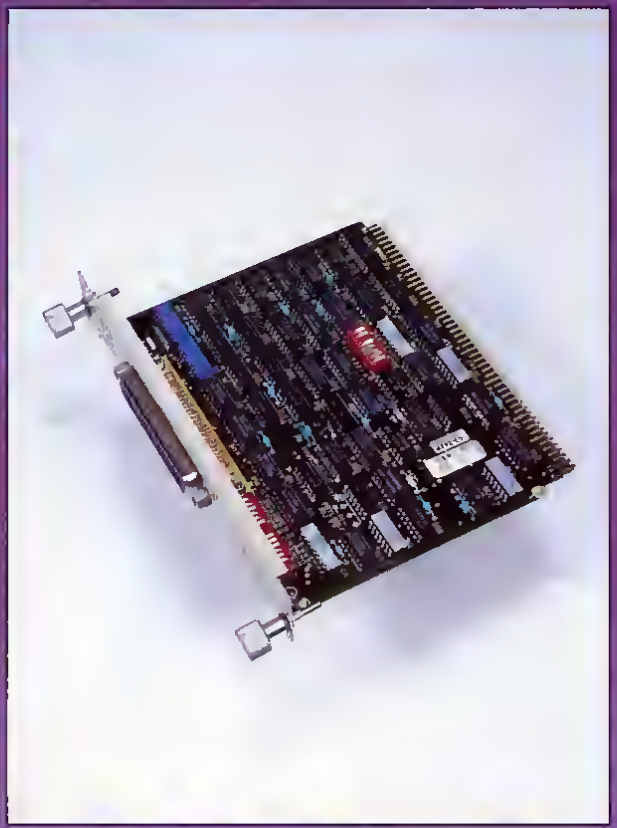
hors série

Led MICRO

PROGRAMMATION
COURS 2ème CYCLE

COURS
N°39
Suite
2e cycle
N°19



COURS DE PASCAL
la récursivité

COURS DE PROGRAMMATION APPROFONDIE :
mouvements du curseur

dBase III

ISSN 0757-6869

M 1988 - 39 - 18,00 F
3791988018008 00390

MENSUEL AVRIL 1987 - BELGIQUE 139 FB/CANADA 3,75 $/SUISSE 6,75 FS

VOYAGE AU CŒUR DES MICRO-ORDINATEURS

dans la COLLECTION «ETUDES» aux éditions fréquences

une véritable schémathèque

- 128 pages
- 101 schémas
- 34 tableaux

Prix : 150 F

Que ce soit pour concevoir des interfaces ou optimiser un programme (utilisation des périphériques, encombrement mémoire...) «un micro-informaticien performant» doit posséder une bonne connaissance de son matériel.

Ce livre s'adresse donc à tous les électroniciens qui désirent découvrir les différents composants constituant un micro-ordinateur. Articulé autour du microprocesseur Z80, cet ouvrage contient de nombreux schémas (plan mémoire, interfaces série et parallèle, interface clavier, interface vidéo, CAN, CNA ...) qui pourraient être le thème ... de nouvelles extensions.

En vente chez votre libraire et aux Editions Fréquences

BON DE COMMANDE

Je desire recevoir l'ouvrage **L'electronique des micro-ordinateurs»** au prix de **160 F** (150 F + 10 F de port)

Nom .....

Adresse .....

A adresser aux EDITIONS FREQUENCES 1 boulevard Ney, 75018 Paris

Règlement ci joint

Par chèque bancaire ☐ par chèque postal ☐ par mandat ☐

Philippe Faugeras, Docteur-ingénieur en électronique a acquis son expérience dans de grandes entreprises françaises où pendent cinq ans, il a travaillé sur des systèmes d'automatismes à base de microprocesseurs. Philippe Faugeras est responsable de la rubrique «Raconte-moi la micro-informatique» dans la revue LED.

hors série
Led MICRO
PROGRAMMATION COURS 2e CYCLE

Societe editrice
Editions Fréquences
Siege social
1 bd Ney 75018 Paris
Tel (1) 46 07 01 97 +
SA au capital de 1 000 000 F
President Directeur General
Edouard Pastor

LED MICRO
(cours 2e cycle)
Mensuel 18 F
Commission paritaire 64949
Directeur de la Publication
Edouard Pastor
Tous droits de reproduction reserves
textes et photos pour tous pays
LED MICRO est
une marque deposee ISSN 0757-5648

Services Redaction-Publicite-
Abonnements
1 bd Ney 75018 Paris
Tél (1) 46 07 01 97
Lignes groupees

Comité de rédaction
Dominique Chastagnier
Jean-François Coblentz
Charles-Henry Delaleu
Patrick Gueneau

Secretaire de Redaction
Chantal Cauchois

Publicité A la revue
Tél 46 07 01 97
Secretaire responsable
Annie Perbal

Abonnements
10 numéros par an
France 160 F
Etranger 240 F

Realisation
Composition-Photogravure
Edi Systemes

Impression
Berger-Levrault - Nancy

Les Editions Frequences edient
La Nouvelle Revue du Son
Son Video Magazine L Audiophile
Led Micro Zero-VU magazine
Music Video Systemes

COURS N°39 Suite 2e cycle N°19

AVRIL 87

NOTRE COUVERTURE : Une carte d'entrées-sorties 16 bits parallèles possédant toutes les caractéristiques techniques requises pour des applications industrielles. Hormis les lignes de data, elle dispose de toutes les lignes d'état et de status

Electronique digitale ?

Notre temps aura témoigné d'une nouvelle technique, une autre façon de communiquer avec l'électronique digitale.
Philippe Duquasna, professeur chargé de cours au CNAM a su dans cet ouvrage en expliquer clairement les fondements.

Philippe Duquesne, ingénieur electronicien (I S E N) est chargé du cours de microprocesseurs au C N A M de Paris Depuis plus de dix ans, il a pris goût à l'enseignement et il est l'auteur d'un ouvrage didactique sur l'electronique digitale et notamment d'un cours pratique de microprocesseurs Fervent pratiquant du « dialogue» école/industrie, après avoir exercé les fonctions de chef de département électronique chez Burroughs, second constructeur mondial en informatique, il est actuellement chef du service Etudes Electroniques au sein de la direction technique chez Messier Hispano Bugatti (groupe SNECMA) avec, pour principal objectif l'introduction des microprocesseurs dans les trains d'atterrissage

En vente chez votre libraire et aux Editions Fréquences

Bon de commande

Je désire recevoir le livre INITIATION A L'ELECTRONIQUE DIGITALE au prix de 105 F (95 F + 10 F de port)
Adresser ce bon aux EDITIONS FREQUENCES 1, bd Ney 75018 PARIS

Nom Prénom
Adresse

Code postal

Règlement effectué ☐ par C C P
☐ par chéque bancaire
☐ par mandat

COURS DE PASCAL

Dominique Chastagnier
Jean-François Coblentz
Patrick Gueneau

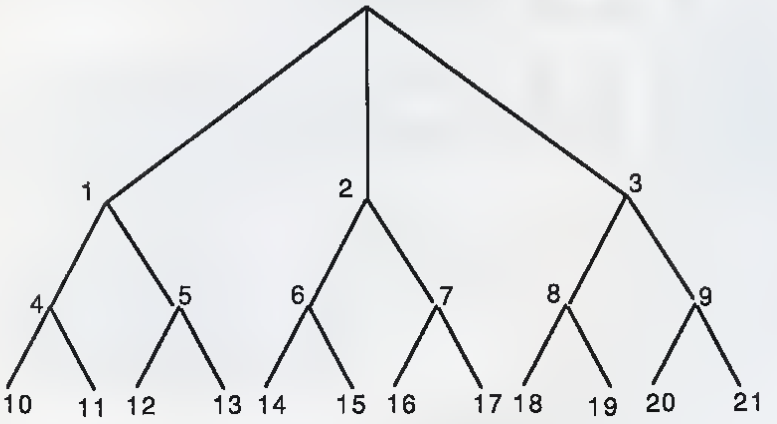
Le mois dernier, nous avons fait un petit tour descriptif de la récursivité. Nous avons montré comment fonctionnait le principe, et surtout que, si cette technique est parfois irremplaçable, elle n'est pas toujours une panacée.
Il nous reste à insister sur ces deux points, fondamentaux lorsque la récursivité est évoquée.
Ce mois-ci, nous allons donner des exemples intéressants d'applications de la récursivité, à travers des techniques qui, pour certaines, sont à la pointe en matière de programmation.

COURS N° 10

PLAN DU COURS
1. Le retour arrière, de l'anglais backtracking
2. Application : les dames en prises
3. Un problème graphique
4. Un exemple mathématique
5. Conclusion
6. Des exercices
   - Les tours de Hanoï
   - Calculer la limite de l'expression
   - Un programme de tri récursif

1. LE RETOUR ARRIERE, DE L'ANGLAIS BACKTRACKING

Voici le probleme Au milieu d'une forêt de possibilites, votre programme doit en choisir une, et pousser au bout ce choix en en faisant d'autres par la suite Si le choix en question est mauvais (on ne parvient pas a la solution), il faut revenir en arriere pour essayer une autre solution Comment faire ? La recursivite, avec son stockage automatique des parametres semble bien adaptee



Sur le graphique representant les choix successifs d'un programme, supposons que la reussite soit l'obtention du chiffre 17 et que le critere de choix soit, par ordre de priorite obligatoire (ce qui veut dire qu'a chaque fois, on essaie la premiere instruction, puis si elle n'est pas possible, la seconde et on recommence ainsi a chaque nouvelle etape) :

- 1 : descendre vers le nœud gauche
- 2 : descendre vers le nœud droit
- 3 : remonter

Dans le cas du graphe precedent, on aurait la suite de nœuds :

R - 1 - 4 - 10 - 4 - 11 - 4 - 1 - 5 - 12 - 5 - 13 - 5 - 1 - R - 2 - 6 - 14 - 6 - 15 - 6 - 2 - 7 - 16 - 7 - 17 - FIN

Il est evident que, au lieu de descendre l'arbre jusqu'au bas de chaque branche, nous aurions pu descendre d'un niveau seulement, et explorer les nœuds de ce niveau : il se serait alors produit la suite de nœuds :

R - 1 - R - 2 - R - 3 - R - 1 - 4 - 1 - 5 - 1 - R - 2 - 6 - 2 - 7 - 2 - R - 3 - 8 - 3 - 9 - 3 - R - 1 - 4 - 10 - 4 - 11 - 4 - 1 - 5 - 12 - 5 - 13 - 5 - 1 - R - 2 - 6 - 14 - 6 - 15 - 6 - 2 - 7 - 16 - 7 - 17 - FIN

Il semble ici que le procede est nettement plus long Si maintenant le critere de succes est de trouver le 3, avec la premiere methode, on aurait :

R - 1 - 4 - 10 - 4 - 11 - 4 - 1 - 5 - 12 - 5 - 13 - 5 - 1 - R - 2 - 6 - 14 - 6 - 15 - 6 - 2 - 7 - 16 - 7 - 17 - 7 - 2 - R - 3 - FIN

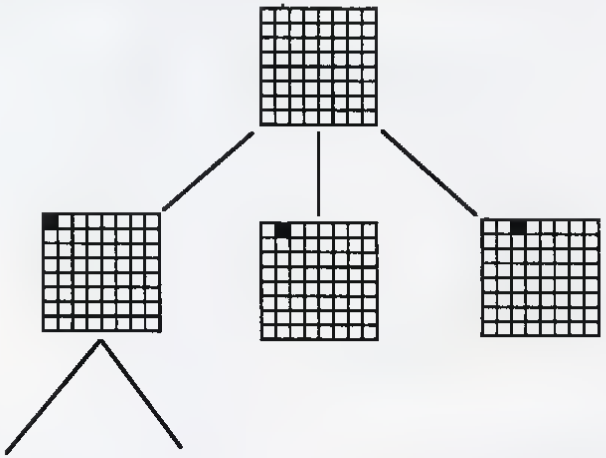
Avec la seconde :

R - 1 - R - 2 - R - 3 - FIN

La preuve est donc faite que chacune des deux methodes a ses avantages et ses inconvenients La premiere est appelee «parcours en profondeur d'abord», la seconde «en largeur d'abord».

Ceci peut vous sembler un peu artificiel Donc, nous allons proposer un jeu, pour clarifier les idees Ce jeu consiste a faire deplacer un cheval sur un echiquier de telle

sorte que toutes les cases soient utilisees une fois seulement Il existe un algorithme mais nous ne le considererons pas, puisque vous ne le connaissez peut-être pas Comment faire ?
Nous pouvons ramener le probleme a la recherche du coup suivant ou, s'il n'existe pas, à l'etude permettant de savoir si on a gagne ou s'il faut recommencer car le jeu est bloque Un exemple graphique d'abord



On le voit, la recherche peut être vraiment longue et delicate Mais, heureusement, l'homme a cree l'ordinateur, et nous allons donc nous en servir !!

2. APPLICATION : LES DAMES EN PRISES

Dans le même ordre d'idées, voici un programme qui execute la recherche complete de toutes les solutions, par une methode recursive du probleme suivant : trouver toutes les positions possibles de n reines sur un echiquier nxn, de telle sorte que les reines ne soient en position de prises sur aucune autre Cela signifie qu'il positionne une piece, puis une autre sur la premiere case libre qu'il trouve, et ainsi de suite S'il se trouve bloque, il revient un cran en arriere, et tente de trouver une autre solution Si c'est impossible, il revient encore un cran en arriere, et ainsi de suite Une alternative est possible, qui arrête le programe des qu'une solution est trouvee Ici, nous choisirons de les trouver toutes Bien sûr, il ne s'agit pas de les imprimer mais de les compter Remarquons que lorsqu'une dame est sur une case, sa ligne et sa colonne ne peuvent plus servir Nous utilisons cette propriete pour debuter sur la case (1,1) la rechercher et augmenter l'indice de ligne et celui de colonne a chaque fois Voici le programme, et les commentaires

```
program Reines;
 const
  dim = 4;
 type
   tab_sol = array[1..dim] of integer;
 var
  tab : tab_sol;
  i, j, nb_sol, k : integer;
  solution : boolean;
```

```
function bloquee (l, c : integer) : boolean;
 var
  ind_l, i : integer;
begin
 bloquee := false;
 for i := 1 to c - 1 do
 begin
   ind_l := abs(tab[i] - l);
   if ind_l * (ind_l - c + i) = 0 then
   bloquee := true;
 end;
end;

procedure trouve_sol (l, c : integer);
 var
  i : integer;
  ok : boolean;
begin
 ok := false;
 if c > dim then
 begin
   nb_sol := nb_sol + 1;
    writeln(nb_sol);
 end
 else
  for i := l to dim do
   if not bloquee(i, c) then
  begin
    tab[c] := i;
    trouve_sol(1, c + 1);
   ok := true;
  end;


  if ((not ok) and (c > 1)) then
  begin
    l := tab[c - 1];
    tab[c - 1] := 0;
    trouve_sol(i + 1, c - 1);
  end;
end;


 procedure init (var i, j, nb_sol : integer;
    var tab : tab_sol);
  var
   k : integer;
begin
 i := 1;
 j := 1;
 nb_sol := 0;
 for k := 1 to dim do
  tab[k] := 0;
end;
```

```
begin
  init(i, j, nb_sol, tab);
  trouve_sol(i, j);
end.
```

Le programme comporte trois parties distinctes .
- Programme principal qui se contente de lancer la recherche, apres une initialisation des donnees
- Recherche des solutions possibles
- Etudes des cases mises en prise lors des placements precedents

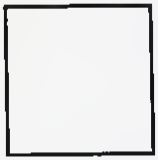
La premiere partie est banale et tres simple. La troisieme est realisee en utilisant une fonction de type booleen qui indique si la case envisagée est déjà bloquee
La seconde partie boucle jusqu a obtention de toutes les solutions, en enchaînant les actions suivantes :
- si *dim* dames sont placées, on a une nouvelle solution (ici dim = 4),
- sinon, on tente d'en placer une nouvelle, a partir des bons indices de ligne et de colonnes,
- on reprend la recherche à partir des conditions du cran précédent

Pour dim = 2, 3, il n'y a pas de solutions. Pour dim = 4, il y en a deux :

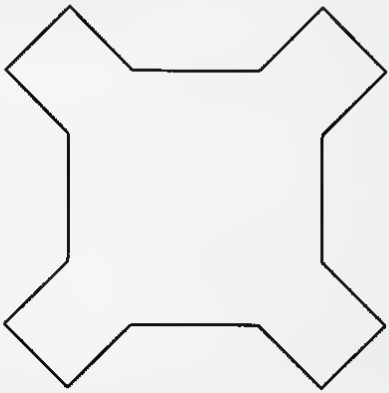
Pour dim = 8, il y a 92 solutions !! Cela commence a en faire un certain nombre, et notre programme travaille alors pendant pres de trois minutes. En comparaison, il faut 19 secondes pour dim = 4

3. UN PROBLEME GRAPHIQUE

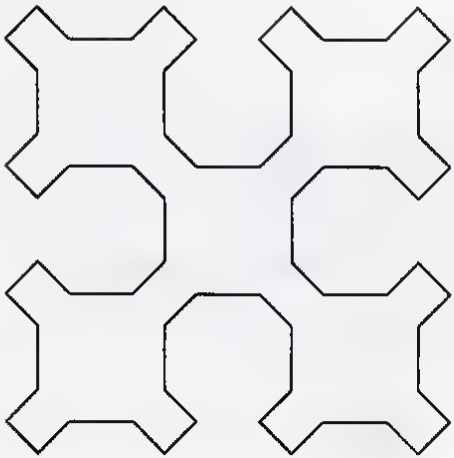
Un probleme completement différent, utilisant les possibilites graphiques des ordinateurs recents. Il est evident que ceux qui ne pourront pas le faire tourner trouveront cela frustrant, mais nous avons prefere le proposer car il illustre avant tout le propos de ce cours. Il est cependant possible d en faire une version reduite en basse resolution. Tout algorithme non recursif pour ce trace est très long, tres complexe. Le programme trace des courbes dites de Sierpinsky. Ce sont des courbes recursives, c'est-a-dire que chaque élément de la courbe dépend des elements precedents.
Le principe est le suivant : considerons l'element graphique de base :



On peut l'utiliser pour former un dessin :



Par une récurrence que nous allons décrire, nous pouvons recommencer cette courbe sur elle-même, et l'on obtient :



On voit que chaque élément est celui de base diminué de moitié, et accroché à l'élément central. On peut à partir de là proposer une définition récursive.

Notons que nous utilisons dans le programme la commande **Forward** qui permet de ne déclarer une procédure qu'après son appel. C'est obligatoire ici, car les procédures A, B, C et D s'appellent les unes les autres, donc il y aurait forcément un problème sans cette possibilité. La commande FORWARD indique simplement au système que la procédure existe, est bien déclarée, mais est décrite plus loin. Cette commande a déjà été décrite, mais vous en voyez ici la première utilisation. Voici le programme :

Ce programme est écrit sur un Macintosh. Il utilise les procédures graphiques de cet ordinateur, à savoir :

- SHOWDRAWING : permet de visualiser la fenêtre graphique,
- MOVETO(x,y) : aller au point x,y, sans rien tracer,
- LINETO(x,y) : aller au point x,y en traçant une ligne depuis le point où l'on se trouve.

Ce programme est simple dans son principe. Il est par contre un peu complexe dans son écriture, l'imbrication entre les procédures en étant la cause.

```
program Sierpinski_LED;
        const
                ordre = 3;
    (* à vous de mettre le nombre que vous souhaitez, mais après 5 c'est long *)
                h0 = 256;
        var
                i, h, x, y, x0, y0 : integer;
        procedure A (i : integer);
                forward;
        procedure B (i : integer);
                forward;
```

```
procedure C (i : integer);
        forward;
procedure D (i : integer);
        forward;
procedure A;
begin
        if i > 0 then
                begin
                        A(i - 1);
                        x := x + h;
                        y := y - h;
                        lineto(x, y);
                        B(i - 1);
                        x := x + 2 * h;
                        lineto(x, y);
                        D(i - 1);
                        x := x + h;
                        y := y + h;
                        lineto(x, y);
                        A(i - 1)
                end
end;

procedure B;
begin
        if i > 0 then
                begin
                        B(i - 1);
                        x := x - h;
                        y := y - h;
                        lineto(x, y);
                        C(i - 1);
                        y := y - 2 * h;
                        lineto(x, y);
                        A(i - 1);
                        x := x + h;
                        y := y - h;
                        lineto(x, y);
                        B(i - 1)
                end
end;

procedure C;
begin
        if i > 0 then
                begin
                        C(i - 1);
                        x := x - h;
                        y := y + h;
                        lineto(x, y);
                        D(i - 1);
                        x := x - 2 * h;
```

```
                    lineto(x,  y);
                    B(i - 1);
                    x := x - h;
                    y := y - h;
                    lineto(x,  y);
                    C(i - 1)
                end
end,

procedure D;
begin
        if i > 0 then
                begin
                    D(i - 1);
                    x := x + h;
                    y := y + h;
                    lineto(x,  y),
                    A(i - 1);
                    y := y + 2 * h;
                    lineto(x,  y);
                    C(i - 1);
                    x := x - h,
                    y := y + h;
                    lineto(x,  y),
                    D(i - 1)
                end
end;


begin
            ShowDrawing;
            i := 0;
            h := h0 div 4;
            x0 := 2 * h;
            y0 := 3 * h;
            repeat
                    begin
                        i := i + 1;
                        x0 := x0 - h;
                        h := h div 2;
                        y0 := y0 + h;
                        x := x0;
                         y := y0;
                        moveto(x,  y);
                        A(i);
                        x := x + h;
                        y := y - h;
                        lineto(x,  y);
                        B(i);
                        x := x - h;
                        y := y - h;
                        lineto(x,  y);
                        C(i);
```

```
                                x := x - h;
                                y := y + h;
                                lineto(x,  y);
                                D(i);
                                x := x + h;
                                y := y + h;
                                lineto(x,  y);
                        end;
                until i = ordre
end.
```

4. UN EXEMPLE MATHEMATIQUE

Il en fallait un, ne nous le reprochez pas. Nous allons donner un exemple court : la génération de nombres premiers. Un nombre est premier s'il ne peut être divisé que par 1 et par lui-même. A partir de cette definition, il vient le programme suivant .

```
program  Premier_recursif;
 var
  p : integer;

 function teste (n, p : integer) : boolean;
 begin
  if sqr(n) > p then
   teste := true
  else
   begin
    if p mod n = 0 then
     teste := false
    else if n > 2 then
     teste := teste(n + 2, p)
    else
     teste := teste(n + 1, p);
   end;
 end;

begin
 repeat
  write('introduire un entier : ');
  readln(p);
 if p <> 0 then
  begin
    if teste(2, p) then
     writeln('ce nombre est premier')
   else
    writeln(' ce nombre n est pas premier ');
    writeln;
    writeln;
  end;
 until p = 0;
end.
```

Ce programme calcule donc si un nombre est premier ou non, en le divisant par 2 et par tous les nombres impairs inferieurs à sa racine carree. Ce critere est suffisant puisque

si un nombre plus grand que sa racine carrée le divise, un nombre plus petit le divisera aussi. En effet, soit p le nombre. S'il existe b, plus grand que $\sqrt{p}$, qui divise p, alors il existe a tel que :

$$a \cdot b = p$$

Si a est plus grand que $\sqrt{p}$, a b est forcement plus grand que p, donc a est plus petit que p et la recherche l'aura trouvée avant de trouver b. Pour un nombre p, on regarde s'il est divisible par 2, puis par 3, puis par la suite des nombres impairs. Les nombres pairs sont éliminés par le fait que si un nombre pair divise p, 2 divise p. Il est possible d'en faire autant avec les multiples de 3, 5, ... mais la relation donnant les nombres a tester devient plus compliquée.

5. CONCLUSION

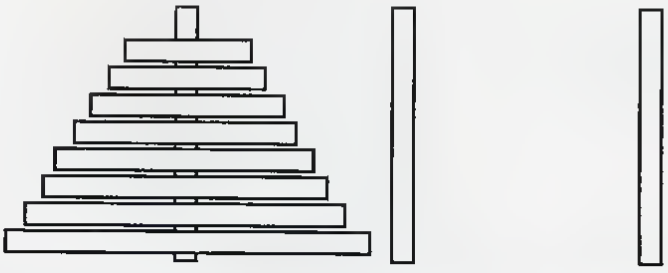
On le voit, la notion de récursivité est d'une grande importance, tant pratique que théorique. Elle est a la base de nombreux algorithmes très importants, tels celui dit de «backtracking» ou de retour arrière. C'est l'algorithme utilisé pour le programme des dames sur l'échiquier. Il consiste a avancer dans la solution tant que c'est possible, puis, en cas de blocage, de revenir en arrière d'un cran, et de recommencer la recherche depuis ce point, dans une autre direction. Cet algorithme est particulièrement utilisé dans les applications actuelles en intelligence artificielle. Il est, en règle générale, très commode pour la rédaction de programmes clairs. Par contre, il devient plus délicat de suivre le déroulement précis du programme. Tout comme la récursivité, il est surtout utile lorsque la solution d'un problème n'est pas directement accessible par un algorithme. Ce qui signifie que la récursivité est particulièrement pratique lorsqu'on n'a aucune idée de ce que l'on cherche.

Le mois prochain, nous aborderons une autre notion capitale en Pascal, les pointeurs, et l'allocation dynamique de mémoire. Ces éléments forment la pierre angulaire des langages évolués et leur implantation première a été faite pour Pascal, puis sur d'autres, mais avec plus ou moins de succès quant à la facilité d'emploi ou la gestion de la mémoire.

6. DES EXERCICES

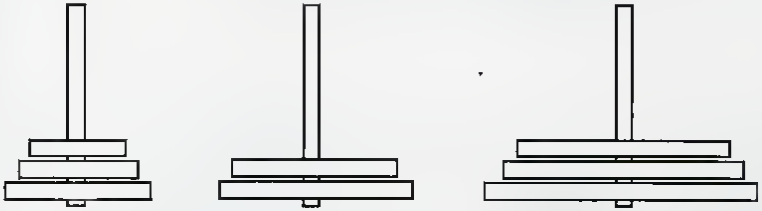
1. Les tours de Hanoï

Voila un problème passionnant, dérivé d'une très saine occupation des moines d'un monastère vietnamien. Vous disposez de trois poteaux et de n disques percés, tous de taille différente. Vous pouvez placer un disque à chaque fois et ne le reposer que sur un disque plus grand. Si vous partez avec tous les disques sur un poteau d'un côté, il faut les amener tous sur le poteau opposé.
La position de départ :

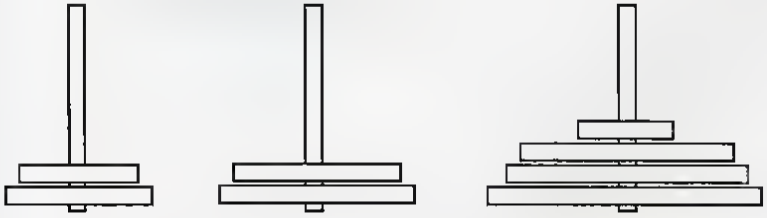


et la position d'arrivee :

Un mouvement possible : de la position

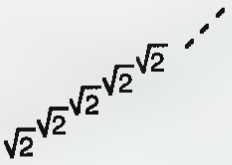


a la position :



L'exercice est de programmer cet amusant jeu a l'aide d'un algorithme récursif, pour un nombre de disque qui sera mis en constante dans le programme, ou demande au début de l'execution.
Un detail : les moines deplacent 64 disques, et la legende veut que lorsque ce travail sera termine, le monde verra sa fin. Il faut dire que dans le cas de 64 disques, le temps (manuel !) est tout à fait considerable. Avec 8 disques, un Pascal compile demande une bonne demi-douzaine de minutes.

2. Calculer la limite de l'expression :



autrement dit, racine de deux, à la puissance racine de deux,...

3. Un programme de tri recursif
Une methode de tri consiste a couper la liste en deux, au centre, puis à refaire de même sur chaque sous-liste, jusqu'a ce qu'il n'y ait qu'un ou deux elements dans la liste courante que l'on ordonne, puis on reunit ces listes deux par deux, et on les ordonne, etc. Faire le programme correspondant. Un exemple sur cinq valeurs :

```
5 2 1 4 3
5 2 1       4 3
5 2         1         3 4
2 5         1 3 4
2 5         1 3 4
2 5 1 3 4
1 2 3 4 5
```

Bonne chance.

COURS DE PROGRAMMATION APPROFONDIE

Dominique Chestagnier
Jean-François Coblentz
Patrick Gueneau

Nous reprenons l'etude des mouvements du curseur de l'éditeur de texte à l'endroit où nous l'avions laissé le mois dernier. Nous y avons inclus le descriptif en pseudo-langage des routines qui s'y rapportent. Il vous faudra encore un peu de patience pour venir à bout de ce projet. Il nous reste en effet à décrire quelques routines essentielles et surtout à effectuer l'assemblage des divers éléments pour enfin obtenir l'application souhaitée (on peut l'esperer).

COURS N° 19

PLAN DU COURS
4. Mouvements du curseur (suite)

4. MOUVEMENTS DU CURSEUR (SUITE)

GAUCHE

```
début_procedure
    X = X-1 , (un cran à gauche)
    si (X=0) alors

        (on est sur le bord de l'ecran)

debut_si
    Y = Y-1 , (il faut passer a la ligne precedente)
    si (Y=0) alors

        (on est en haut il faut donc decaler)

    debut_si
        DEF_HAUT_BAS(err) ,
        Y = 1 ,
    fin_si
    si (err est faux) alors

        (si on n'est pas en haut du buffer)
        (il faut donc bouger le curseur)

    debut_si
          (on se replace en fin de ligne)
          X = long (Buffer(prem_ligne + Y - 1) ) ,
                    sinon
          X = 1 ;
          ERREUR ; (beep et pas de mouvement)
        fin_si
    fin_si
fin_procedure
```

DROITE

```
debut_procedure
    X = X+1 , (un cran a droite)
    si (X > long (buffer(prem_ligne + Y - 1) ) ) alors

        (on est a la fin de la ligne)
    début_si
        Y = Y+1 ; (il faut passer a la ligne suivante)
        si (Y > nb_ligne) alors

        (on est en bas, il faut donc decaler)

        début_si
          DEF_BAS_HAUT(err) ;
          Y = nb_ligne ,
        fin_si
        si (err est vrai)      alors

        (si on est en bas du buffer)
        (il faut donc reprendre l'ancienne place du curseur)

        debut_si
          X = X-1 , (on se replace en fin de ligne)
          ERREUR , (beep et pas de mouvement)
```

```
                                sinon
                    X = 1 , (on est en debut de ligne suivante)
                fin_si
            fin_si
fin_procédure
```

Remarque sur GAUCHE ET DROITE :
On s aperçoit que les routines DEF_HAUT_BAS et DEF_BAS_HAUT ne traitent pas directement l erreur Il faut donc le faire dans les routines appelantes L'avantage de cette solution est de differencier les operations a effectuer en retour de l appel - nous verrons pour les procedures HAUT et BAS, l interet de ce traitement differe

MONTE

```
début_procedure
    Y = Y - 1 ;
    si (Y = 0) alors

        (il faut directement appeler DEF_HAUT_BAS)

    début_si
        DEF_HAUT_BAS(err) ,
        Y = 1 ;          (on reste en haut de l ecran)
        si (err est vrai) alors

        (on ne fait nen et on genere une erreur)

        début_si
          ERREUR ;
        fin_si
    fin_si

        (calcul de la nouvelle position du curseur)

    X0 = long (buffer(prem_ligne + Y - 1) ) ,
    si (X > X0) alors

        (il faut mettre alors le curseur en fin de ligne)
        (c est en fait une convention mais elle simplifie la gestion)

    début_si
        X = X0 + 1 ;          (apres le dernier caractere)
    fin_si
fin_procedure
```

DESCENDRE

```
début_procédure
    Y = Y + 1 ;
    si (Y > nb_ligne) alors

        (il faut directement appeler DEF_BAS_HAUT)

    début_si
        DEF_BAS_HAUT(err) ,
        Y = nb_ligne          (on reste en bas de l écran)
        si (err est vrai) alors

        (on ne fait nen et on genere une erreur)
```

```
        debut_si
          ERREUR ,
        fin_si
    fin_si

        (calcul de la nouvelle position du curseur)

    X0 = long (buffer(prem_ligne + Y - 1) ) ,
    si (X > X0) alors

        (il faut mettre alors le curseur en fin de ligne)
        (c est en fait une convention, mais elle simplifie la gestion)

    début_si
        X = X0 + 1 ;          (apres le dernier caractere)
        fin_si
fin_procédure
```

(Rappel : nb_ligne est le nombre de lignes de l'ecran et prem_ligne le n° de la premiere ligne affichee du buffer.)

Il ne nous reste plus qu'à ecrire les deux routines de defilement.

```
DEF_HAUT_BAS :

début_procédure
    prem_ligne = prem_ligne - 1 ;
    si (prem_ligne = 0) alors

        (on est en haut du buffer donc on ne fait rien)

    début_si
        prem_ligne = 1 ;          (on restitue l'ancienne valeur)
        err = vrai ;
                    sinon
        err = faux

    (on affiche tout l'ecran)

        effacement ;          (routine standard d effacement d ecran)

    (boucle sur les nb_ligne lignes a afficher)

        pour i allant_de prem_ligne à (prem_ligne + nb_ligne - 1)

        début_pour
          position_ecran(1, i - prem_ligne + 1) ,
          écrit(buffer(i) ) ,
        fin_pour

    fin_si

DEF_BAS_HAUT :

début_procédure
    prem_ligne = prem_ligne + 1 ,
    si (prem_ligne > nb_occup) alors

        (on est bas du buffer donc on ne fait rien)
```

```
début_si
    prem_ligne = nb_occup ;      (on restitue l'ancienne valeur)
    err = vrai ;
                                 sinon
    err = faux ;

    (on affiche tout l'écran)

    effacement ;      (routine standard d'effacement d'ecran)

    (boucle sur les nb_ligne lignes à afficher)

    pour i allant_de prem_ligne à (prem_ligne + nb_ligne - 1)

    début_pour
      position_ecran(1, i - prem_ligne + 1)
      ecrit(buffer(i) ) ;
    fin_pour

fin_si
```

rappel : nb_occup est le nombre total de lignes utilisees du buffer

CONCLUSION

Il nous reste donc a etudier quelques routines comme celles de destructions (caracteres, lignes), mais aussi celle fondamentale d'insertion d'une nouvelle ligne ; elle sera affectee bien evidemment a la touche «return». C'est cette derniere procedure qui sera notamment chargee de detecter un dépassement de capacite du buffer. Nous verrons aussi qu'elle provoque de nombreuses operations tant sur ce buffer que sur l'affichage (defilement partiel de l'écran, modification des lignes, etc.) Nous pourrons ensuite decrire en detail le fonctionnement général du programme (nous fournirons le programme source en Pascal et en Basic de la routine CORPS) Enfin, nous reviendrons quelque peu sur les details des differentes routines que nous presentons ce mois-ci pour permettre de mieux comprendre le comment et le pourquoi de nos choix

DIALOGUE AVEC NOS LECTEURS

Ce mois-ci, nous ne traiterons que la résolution du troisième degré. Nous l'avons adaptée à un problème mathématique souvent rencontré dans des applications pratiques : la détermination des valeurs propres et vecteurs propres d'une matrice 3×3 symétrique. Pour des raisons de place, nous ne vous soumettrons que la solution Pascal, mais la traduction en Basic n'est pas d'une grande difficulté.

I. VALEURS PROPRES ET VECTEURS PROPRES

Nous vous prions de nous excuser du caractère exagérement «matheux» de cet exercice et nous bornerons à rappeler les propriétés des vecteurs propres et valeurs propres, considérant que la théorie mathématique sous-jacente n'est pas de notre domaine.

Toute matrice M peut se mettre sous la forme d'un produit de trois matrices :

$$M = P^{-1} A P \qquad \text{ou } P^{-1} \cdot P = I$$

I est la matrice identité car elle a des zéros partout sauf sur la diagonale où il y a des 1 et le produit de M . I = M.

A est une matrice nulle sauf sur la diagonale où se trouvent toutes ses valeurs propres.

Pour déterminer les valeurs propres d'une matrice, on résoud l'équation $M - \lambda I = 0$, équation en λ du degré de la taille de la matrice.

Ici, c'est bien notre équation du troisième degré.

Après cela, à chaque valeur propre correspond un vecteur propre, vecteur qui reste invariant à la transformation par $M - \lambda I$ ou λ_i est la $i^{ème}$ valeur propre.

II. MISE EN ŒUVRE

1. Quelques remarques préliminaires

Ce programme a été prévu pour calculer vite et bien, c'est-à-dire qu'il ne peut se permettre des résultats fantaisistes ni pousser très loin la précision, ceci nécessitant généralement beaucoup trop de calculs.

Ainsi, on ne teste pas par rapport à 0 mais à un epsilon donné. De plus, la résolution du troisième degré est hybride entre la solution purement algébrique et la méthode trigonométrique.

2. Commentaires de programmation

– Il manque de commentaires ! Exemple-type de programme de «matheux».
– On détermine l'équation $\lambda^3 + B\lambda^2 + C\lambda + D$ d'où l'on sort $x^3 + px + q$.
– La matrice est symétrique donc toutes les racines sont réelles.
– Le type REEL = REAL peut faire rire mais si vous voulez passer en double précision, vous n'avez qu'à le changer une seule fois dans le programme, c'est très agréable.

```
{ Module de calcul de vecteurs propres et valeurs propres }

PROGRAM CALCUL_VECTEURS ;


Const

  d_pi_s_3 = 2.0943951 ; { deux * pi / trois }
  epsilon = 0.00001 ;    { zero de precision }
  n_vol_max = 100 ;


Type
  solutions = (simples,doubles,triples) ;

  REEL = REAL ;

  T_MATRICE = ARRAY [1..3,1..3] OF REEL ;
  T_VECTEUR = ARRAY [1..3] OF REEL ;

  T_REEL_VOL    = ARRAY [0..N_VOL_MAX] OF REEL          ;
  T_INTEGER_VOL = ARRAY [0..N_VOL_MAX] OF INTEGER       ;

  TEXTE = ARRAY [1..8] OF CHAR ;

Var

  ok : boolean ;
  i  : integer ;
  matrice , vect_prop : T_MATRICE ;
  val_prop  : T_VECTEUR ;

{-------------------------------------------------------------------------------}

Function Arc_cos ( X : REEL ) : REEL ;

  Begin
    Arc_cos := ARCCOS ( X ) ;
  End ;

{-------------------------------------------------------------------------------}

Function Cub_root ( X : REEL ) : REEL ;

 { Calcul de la racine cubique d' un reel ;
   si celui-ci est negatif il rend la valeur negative }

  Var
    signe : BOOLEAN ;
    result : REEL ;

  Begin
    signe := false ;
    IF X < 0 THEN
      begin
        X := - X ;
        signe := true ;
      end ;
    result := EXP ( LN ( X ) / 3.0 ) ;
    IF signe THEN result := - result ;
    Cub_root := result ;
  End ;

{-------------------------------------------------------------------------------}
```

```
Function Valeurs_propres ( IN  matr_A       : T_MATRICE ;
                           OUT x            : T_VECTEUR ;
                           OUT val_propres : solutions ) : BOOLEAN ;

  Var

    i , j : integer ;
    B , C , D , p , q , racine , theta , discriminant , test : REEL ;
    ok : BOOLEAN ;
    matr_t : ARRAY [1..6,1..6] OF REEL ;

  Begin
    ok := true ;
    for i := 1 to 3 do
        for j := 1 to 3 do
          begin
            matr_t[i,j] := matr_A[i,j] ;
            matr_t[i+3,j] := matr_A[i,j] ;
            matr_t[i,j+3] := matr_A[i,j] ;
            matr_t[i+3,j+3] := matr_A[i,j] ;
          end ;
    B := 0.0  ;
    for i := 1 to 3 do B := B - matr_t[i,i] ;
    C := 0.0 ;
    for i := 1 to 3 do
        C := C + matr_t[2 * i - 1,2 * i - 1] * matr_t[2 * i,2 * i]
               - matr_t[i + 1,i] * matr_t[i,i + 1] ;
    D := 0.0 ;
    for i := 1 to 3 do
      begin
        p := 1 ;
        for j := 1 to 3 do
            p := p * matr_t[i + j - 1,j] ;
        D := D - p ;
        p := 1 ;
        for j := 3 downto 1 do
            p := p * matr_t[i - j + 3,j] ;
        D := D + p ;
      end ;
    p := ( 3.0 * C - SQR ( B ) ) / 3.0 ;
    q := ( 2.0 * B * B * B - 9.0 * B * C + 27.0 * D ) / 27.0 ;
    discriminant := q * q / 4.0 + p * p * p / 27.0 ;
    IF ABS ( q ) < 1 THEN test := epsilon
                     ELSE test := 0.0000001 * q * q ;
    IF discriminant > test THEN ok := false

    ELSE IF discriminant > - test THEN
      begin
        racine := 2 * cub_root( - q / 2.0 ) ;
        B := - B / 3.0 ;
        x[1] := racine + B ;
        x[2] := ( - racine / 2.0 ) + B ;
        x[3] := x[2] ;
        IF racine < test THEN val_propres := triples
                         ELSE val_propres := doubles ;
      end

    ELSE   { discriminant > 0 }
      begin
        racine := 2.0 * SQRT( - p / 3.0 ) ;
        theta := arc_cos( 3.0 / p * q / racine) / 3.0 ;
        x[1] := racine * COS( theta ) - B / 3.0 ;
        x[2] := racine * COS( theta + d_pi_s_3 ) - B / 3.0 ;
        x[3] := racine * COS( theta + 2 * d_pi_s_3 ) - B / 3.0 ;
        val_propres := simples ;
      end ;
```

```
    Valeurs_propres := ok ;
  End ;


{-------------------------------------------------------------------------------}


Function Calcul_vect_propres (    matr_A    : T_MATRICE ;
                              OUT val_prop  : T_VECTEUR ;
                              OUT vect_prop : T_MATRICE ) : BOOLEAN ; EXTERN ;



  Var
    i , j , k : integer ;
    determinant , norme , verif , seuil : REEL ;
    val_propres : solutions ;
    matr_t : T_MATRICE ;
    ok : BOOLEAN ;

  Begin
    ok := valeurs_propres ( matr_A , val_prop , val_propres ) ;
    seuil := 0.0 ;
    for i := 1 to 3 do seuil := seuil + ABS ( val_prop[i] ) ;
    seuil := 0.0001 * seuil ;
    IF val_propres = simples THEN
        for i := 1 to 3 do
          begin
            for j := 1 to 3 do
                for k := 1 to 3 do
                    matr_t [ j , k ] := matr_A [ j , k ] ;
            for k := 1 to 3 do
                matr_t [ k , k ] := matr_t [ k , k ] - val_prop [ i ] ;
            vect_prop [ 3 , i ] := 1.0 ;
            determinant := matr_t [ 1 , 1 ] * matr_t [ 2 , 2 ]
                         - matr_t [ 2 , 1 ] * matr_t [ 1 , 2 ] ;
   IF ABS ( determinant ) > epsilon THEN
     begin
       vect_prop [ 1 , i ] := ( matr_t [ 2 , 3 ] * matr_t [ 1 , 2 ]
                              - matr_t [ 1 , 3 ] * matr_t [ 2 , 2 ] ) / determinant ;
       vect_prop [ 2 , i ] := ( matr_t [ 1 , 3 ] * matr_t [ 2 , 1 ]
                              - matr_t [ 2 , 3 ] * matr_t [ 1 , 1 ] ) / determinant ;
       verif := 0.0 ;
       for j := 1 to 3 do verif := verif + matr_t [ 3 , j ] * vect_prop [ j , i ] ;
     end
   ELSE
       verif := 1.0 ;
   IF ABS ( verif ) > seuil THEN
     begin
       vect_prop [ 2 , i ] := 1.0 ;
       determinant := matr_t [ 1 , 1 ] * matr_t [ 3 , 3 ]
                    - matr_t [ 3 , 1 ] * matr_t [ 1 , 3 ] ;
       IF ABS ( determinant ) > epsilon THEN
         begin
           vect_prop [ 1 , i ] := ( matr_t [ 3 , 2 ] * matr_t [ 1 , 3 ]
                                  - matr_t [ 3 , 3 ] * matr_t [ 1 , 2 ] ) / determina
           vect_prop [ 3 , i ] := ( matr_t [ 1 , 2 ] * matr_t [ 3 , 1 ]
                                  - matr_t [ 3 , 2 ] * matr_t [ 1 , 1 ] ) / determina
           verif := 0.0 ;
           for j := 1 to 3 do verif := verif + matr_t [ 2 , j ] * vect_prop [ j , i
         end ;
     end ;
   IF ABS ( verif ) > seuil THEN
     begin
       vect_prop [ 1 , i ] := 1.0 ;
       determinant := matr_t [ 2 , 2 ] * matr_t [ 3 , 3 ]
```

```
                    - matr_t [ 3 , 2 ] * matr_t [ 2 , 3 ] ;
      IF ABS ( determinant ) > epsilon THEN
        begin
          vect_prop [ 2 , i ] := ( matr_t [ 3 , 1 ] * matr_t [ 2 , 3 ]
                                 - matr_t [ 3 , 3 ] * matr_t [ 2 , 1 ] ) / determina
          vect_prop [ 3 , i ] := ( matr_t [ 3 , 1 ] * matr_t [ 2 , 2 ]
                                 - matr_t [ 3 , 2 ] * matr_t [ 2 , 1 ] ) / determina
          verif := 0.0 ;
          for j := 1 to 3 do verif := verif + matr_t [ 1 , j ] * vect_prop [ j , i
        end ;
    end ;
      IF ABS ( verif ) < seuil THEN
        begin
          norme := 0.0 ;
          for j := 1 to 3 do norme := norme + SQR ( vect_prop [ j , i ] ) ;
          norme := SQRT ( norme ) ;
          for j := 1 to 3 do vect_prop [ j , i ] := vect_prop [ j , i ] / norme ;
        end
      ELSE
        begin
          writeln (' pas de vecteur propre ') ;
          ok := false ;
        end ;
    end
ELSE IF val_propres = doubles THEN
  begin
    for j := 1 to 2 do
        for k := 1 to 3 do
            matr_t [ j , k ] := matr_A [ j , k ] ;
    vect_prop [ 3, 1 ] := 1.0 ;
    for k := 1 to 2 do
        matr_t [ k , k ] := matr_t [ k , k ] - val_prop [ 1 ] ;
    determinant := matr_t [ 1 , 1 ] * matr_t [ 2 , 2 ]
                 - matr_t [ 2 , 1 ] * matr_t [ 1 , 2 ] ;
    vect_prop [ 1 , 1 ] := ( matr_t [ 2 , 3 ] * matr_t [ 1 , 2 ]
                           - matr_t [ 1 , 3 ] * matr_t [ 2 , 2 ] ) / determinant ;
    vect_prop [ 2 , 1 ] := ( matr_t [ 1 , 3 ] * matr_t [ 2 , 1 ]
                           - matr_t [ 2 , 3 ] * matr_t [ 1 , 1 ] ) / determinant ;
    for k := 1 to 2 do
        matr_t [ k , k ] := matr_A [ k , k ] - val_prop [ 2 ] ;
    vect_prop [ 1 , 2 ] := 1.0 ;
    vect_prop [ 1 , 3 ] := 0.0 ;
    vect_prop [ 2 , 2 ] := 0.0 ;
    vect_prop [ 2 , 3 ] := 1.0 ;
    i := 1 ;
    WHILE ( i < 4 ) and ( matr_t[ i , 1 ] * matr_t[ i , 3 ] = 0 ) DO
        i := i + 1 ;
    IF i < 4 THEN
        vect_prop [ 3 , 2 ] := - matr_t [ i , 1 ] / matr_t [ i , 3 ]
    ELSE
        vect_prop [ 3 , 2 ] := 0 ;
    i := 1 ;
    WHILE ( i < 4 ) and ( matr_t[ i , 2 ] * matr_t[ i , 3 ] = 0 ) DO
        i := i + 1 ;
    IF i < 4 THEN
        vect_prop [ 3 , 3 ] := - matr_t [ i , 2 ] / matr_t [ i , 3 ]
    ELSE
        vect_prop [ 3 , 3 ] := 0 ;
    for i := 1 to 3 do
      begin
        norme := 0.0 ;
        for j := 1 to 3 do norme := norme + SQR ( vect_prop [ j , i ] ) ;
        norme := SQRT ( norme ) ;
        for j := 1 to 3 do vect_prop [ j , i ] := vect_prop [ j , i ] / norme ;
      end ;
  end
```

```
  ELSE
    begin
      for i := 1 to 3 do
          for j := 1 to 3 do
              IF i = j THEN vect_prop [ i , j ] := 1.0
                       ELSE vect_prop [ i , j ] := 0.0 ;
    end ;
  Calcul_vect_propres := ok ;
End ;
Begin      { Programme Principal }

  ok := Calcul_vect_propres ( matrice , val_prop , vect_prop ) ;

End.
```

M. Michel Auchère nous a envoyé trois petits programmes de traces de courbes qui sont très intéressants en tant que «benchmarks» graphiques pour évaluer les performances de votre machine

III. BENCHMARKS GRAPHIQUES

I. EPICYCLOÏDES

```
5 REM RAYON DU GALET R
10 R=1
20 INPUT "RAYON DE LA PISTE "; P
30 CLS
40 FOR T=0 TO 2*PI*R STEP 0.02
50 RAD
60 X=(P+R) * SIN(T)-R * SIN(T * (1+P/R))
70 Y=(P+R) * COS(T)-R * COS(T * (1+P/R))
80 PLOT 15 * X, 15 * Y
90 NEXT
100 PLOT 0, -200
110 DRAW 0, 200
120 PLOT -320, 0
130 DRAW 320, 0
140 GOTO 140
```

II. HYPOCYCLOÏDES

```
5 REM RAYON DU GALET R
10 R=1
20 INPUT "RAYON DE LA PISTE "; P
30 CLS
40 FOR T=0 TO 2 * PI * R STEP 0.02
50 RAD
60 X=(P-R) * COS(T)+R * COS(T * (P/R-1))
70 Y=(P-R) * SIN(T)-R * SIN(T * (P/R-1))
80 PLOT 20 X, 20 Y
90 NEXT
100 PLOT 0, -200
110 DRAW 0, 200
120 PLOT -320, 0
130 DRAW 320, 0
140 GOTO 140
```

III. CYCLOÏDES

```
5 INPUT "RAYON "; R
10 CLS
20 PLOT 0, 100
30 DRAW 640, 100
40 FOR T=0 TO 8 * PI * R STEP 0.1
50 RAD
60 X=T-SIN(T)
70 Y=1-COS(T)
80 PLOT 25 * X, 100 * Y + 100
90 NEXT T
100 GOTO 100
```

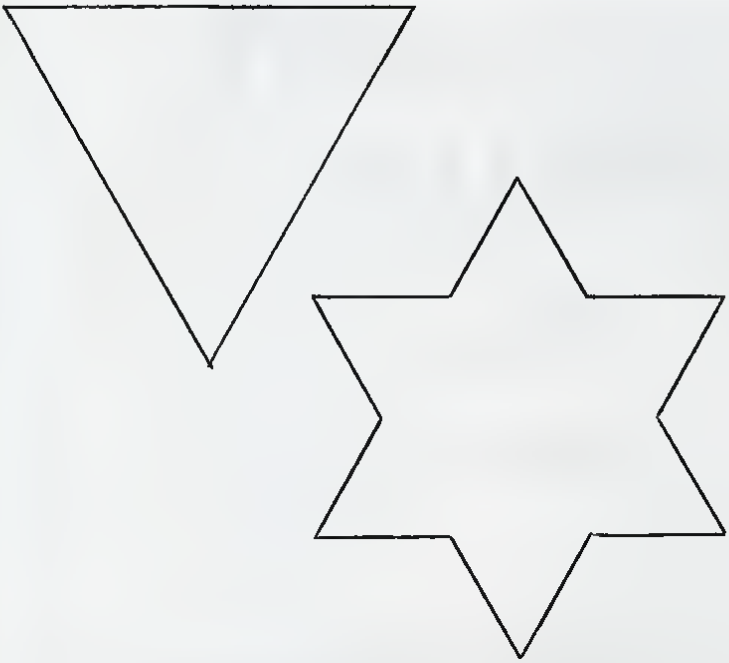
IV. LES FRACTALS

Nous allons aujourd'hui vous presenter un sujet qui passionne beaucoup de scientifiques et autant de non-scientifiques : les Fractals (et non pas «fractaux» car ce nom est issu de l'anglais)

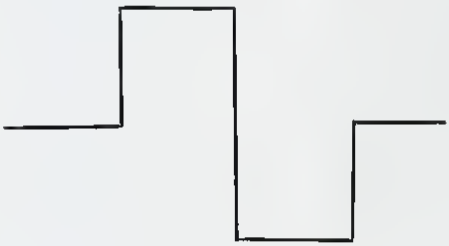
En quoi consiste un fractal ? C'est simplement un segment qui est remplacé par un motif de même longueur que le dit segment : ce motif etant lui-meme uniquement constitué de segments de longueur identique, cette longueur pouvant toutefois differer de la longueur du segment originel.

Une fois effectuee la substitution, on la recommence sur les nouveaux segments

De proche en proche, on obtient un dessin de plus en plus sophistique. Il est alors interessant de definir la figure de départ non pas comme un seul segment mais comme un groupe de segments constituant une figure simple (triangle équilateral, carre, pentagone, etc.) ne possedant comme seule contrainte que la longueur invariante de ses composants



Le dernier sujet d etude reste la definition du modele de substitution Au contraire de ce que l on pourrait croire, les plus complexes ne sont pas toujours les plus jolis Si vous en trouvez un paraissant original envoyez-le nous



Du point de vue programmation, la recursivite du processus est evidente si le programme est aise en BASIC, il devient enfantin en PASCAL. Bon courage

C'EST ARRIVE DEMAIN

(en direct de notre envoyé permanent dans la Silicon Valley)

A l'aide de matériels informatiques somme toute assez rudimentaires, a été menée une étude fort intéressante sur la vigilance des pilotes de lignes lors des grands trajets intercontinentaux. Les conclusions sont à la fois instructives et inquiétantes. Il ressort de ce travail que les pilotes sont en état de sous-vigilance la plupart du temps, en raison de l'inadaptation du corps aux décalages horaires, et surtout à la multiplication de ces décalages dans les deux sens. Jours raccourcis puis allongés sont épuisants pour l'homme. L'étude qui portait sur plusieurs compagnies, montre que la jeunesse des équipages est un critère primordial de sécurité, la baisse de performances étant particulièrement nette après 35 ans. Au palmarès de la sécurité vient d'ailleurs en tête Air New Zealand, qui met ses équipages à la retraite d'office à 40 ans. Les compagnies américaines viennent en queue de classement, car aux USA le principe de la retraite obligatoire est illégal. La France est dans une moyenne que certains trouveront honorables.

Un développeur propose un petit bijou de programme et d'électronique permettant de faire tourner sur un Atari ST la plupart des programmes du MAC. Pour qu'un tel programme tourne sur Atari, il suffit qu'il soit programmé «correctement» sur le Mac, c'est-à-dire qu'il suive à la lettre l'interface Apple. Si c'est le cas, vous aurez le bonheur de le voir tourner sur votre ST et en plus, bien plus vite que sur un Mac. En effet, l'horloge interne du processeur de l'Atari est plus rapide. Comment une telle prouesse est-elle possible ? C'est simple en théorie, puisque les deux ordinateurs ont le même cœur, le 68000. En pratique, c'est moins immédiat et, en particulier, il vous faudra acheter les ROM de l'Apple, le fabricant de l'émulateur ne voulant pas prendre le risque de poursuites judiciaires de la part d'Apple. Mais ces ROM sont disponibles pour une trentaine de dollars chez un bon nombre de détaillants. En tout, pour moins de deux cents dollars, vous aurez un Mac en plus de votre Atari. Atari façon, désolé, n'est pas une riche.

Voilà une idée simple, dans sa formulation tout au moins, qui pourrait bien être la clé de la bureautique du futur proche. Borland va prochainement proposer un traitement de texte qui permet de choisir son mode de fonctionnement. Cela signifie que les usagers habituels de Wordstar pourront choisir un environnement Wordstar complet, et il en va de même pour la plupart des grands traitements de textes populaires (Wordstar, Word Perfect, Word et Multimate dans un premier temps). Il est clair que c'est là une bonne idée puisque cela permet à une société d'acquérir ce programme sans s'inquiéter de la formation qu'ont reçue

ses employes Ils connaissent tous l'un au moins de ces programmes Ce produit est prevu pour les compatibles PC, cela va de soi, puisque sur les autres machines, Mac, Amiga et Atari l'interface est figee plus ou moins par la machine elle-même et le temps d adaptation est nettement raccourci

Une petite revolution vient de se produire sur le marche des imprimantes laser L un des deux fabricants de cette technologie Canon vient de mettre au point un procede permettant de realiser une imprimante pour la moitie du prix des materiels actuels Seule contrepartie, le temps d'impression est moins bon, de l ordre de 6 pages par minute au lieu de 8 actuelles Mais cela permet de mettre en vente des machines valant moins de 2 000 $ au lieu des 4 000 a 5 000 du moment L autre fabricant, Ricoh, propose deja des machines dans ces gammes de prix, ayant egalement une vitesse de 6 pages par minute, mais pour une resolution moins bonne Il s'agit donc bien d'un nouveau pas franchi vers une imprimante laser bon marche, au moment où la demande se fait de plus en plus grande pour ces substituts extrêmement economiques de l impression traditionnelle Une imprimante a laser pour 2 000 $, cela n est pas trop cher comparativement a une imprimante traditionnelle qualite courrier qui n en vaut pas loin de 1 000

C'est l epoque des annonces de nouvelles machines Tout comme le Beaujolais, l ordinateur nouveau est arrive chez tous les fabricants Tous ou presque En effet, ATT, grand rival americain d IBM sur le creneau des compatibles, doit renoncer a la sortie de sa nouvelle gamme pour cause de non compatibilite On pourrait croire que ce type de ridicule peripetie ne pourrait se produire que chez Bull Pas du tout Mais, il est quand même surprenant qu un tel fabricant de compatibles bien rode, rencontre un probleme aussi grotesque, peu de jours avant la mise en vente de sa nouvelle gamme En fait la compatibilite de ses produits serait de l ordre de 60 %, ce qui est absolument inacceptable sur un marche ou les vrais compatibles sont legions

Pour parler des autres nouveautes, en voici une rapide liste des principales Atari sort un PC pour 500 $, disposant de la compatibilite avec le XT, le milieu de gamme IBM Seule limitation reelle, il n a pas de possibilite d extension, un peu comme l Atari ST Compaq propose une nouvelle version de portable, plus leger et un peu plus puissant, utilisant un processeur 80286, le processeur a la mode pour les nouveaux PC Apple propose un «file server» base sur Appletalk le reseau maison, permettant egalement de se connecter sur un reseau de PC Un seul probleme mais un peu gros, il demande un Mac uniquement pour le faire tourner ce qui signifie quand même un systeme conséquent pour un réseau Grid a montre la semaine derniere un portable vraiment beau et puissant, compatible AT, disposant en standard de plus de 1 Mo et d un disque 3,5 pouce de 720 Ko Le tout pour moins de 1 700 $, ce qui est competitif Enfin IBM annonce trois nouvelles machines, dont un bas de gamme au prix de 700 $, ce qui est plus cher que les milieu-haut de gamme de certains compatibles et deux autres ordinateurs tres interessants Le premier est base sur le 80286 et devrait remplacer le AT sous peu, il dispose en standard de la couleur et possede tous les plus actuels, dont les meilleurs systemes d exploitation du moment pour PC, en memoire morte Il sera helas, sans doute très cher, trop peut-être pour être competitif Enfin, IBM propose un appareil base sur le 80386, la derniere petite merveille d'Intel, ultra-rapide et permettant de gerer une très importante memoire Ce dernier sera pres dans peu de temps, mais peu de details sont encore connus

Le Mac s est taille pour l instant la part du lion dans un domaine nouveau, la micro-impression, traduction personnelle et libre de «Desktop Publishing» Ses possibilites graphiques liees a un systeme d exploitation simple, le rendaient particulierement apte a ce type d'applications Mais, devant le developpement important du domaine, les compatibles PC entrent aussi sur le marche En particulier, un programme realise par un developpeur habitue au Mac, Aldus Corporation, permet d en faire autant ou presque qu'avec le même produit sur le Mac Des ameliorations ont même ete apportees, qui se retrouveront bientôt sur la version Mac Mais surtout, ce type de programmes peut tirer un grand profit de la vitesse des PC-AT et autres haut de gamme des compatibles PC Ils utilisent en general Windows, une interface logicielle de Microsoft, qui transforme un PC en sorte de Mac Windows est lent mais, sur un AT, cela n est pas trop sensible Et sur un PC, vous pouvez avoir la couleur Tous ces programmes sont disponibles et il faudra debourser 700 $, ce qui est enorme pour un particulier, mais peu pour une petite societe qui evitera ainsi les frais d impression eleves pour des brochures simples

Il se propage une rumeur intéressante ces derniers temps IBM proposerait bientôt une nouvelle gamme de PC Rien de bien passionnant, pensez-vous Erreur, car cette nouvelle gamme ne serait pas PC-compatible Le monde a l envers Alors que croire ? Il semble que la realite soit bien mesquine une fois de plus A chaque fois qu une rumeur semble prendre sa source chez IBM, c'est pour aider cette societe a prendre des parts de marche ou tout au moins en perdre le moins possible avant la sortie d'un nouveau materiel La derniere fois, c'etait avant la sortie du PC Jr Tout le monde savait ou croyait savoir qu'un merveilleux ordinateur familial allait sortir et qu il fallait attendre plutôt qu acheter un autre produit A cette epoque, la montagne avait accouche de la souris que

l'on sait, et le Junior fut un échec retentissant. Aujourd'hui, la plupart des sociétés de grande et moyenne tailles semblent vouloir renouveler leur parc de compatibles PC, car les progrès en termes de vitesses et de puissances ont été importants en un an. Que fait IBM ? Il leur dit d'attendre, par le biais de cette rumeur. Mais aussi, une contre-rumeur maléfique se propage. Elle dit que le futur PC sera totalement incompatible non seulement avec les PC mais aussi avec les extensions existantes. Ceci veut dire qu'IBM souhaite re-établir un monopole de fait sur les ordinateurs et sur les extensions en tous genres. En bref, tenter de réussir ce qui a été manqué la première fois, mais cette manœuvre attaque de front toutes les sociétés qui se sont équipées de matériels lourds et coûteux sous la foi de la stabilisation qu'apportait le standard actuel PC, de même que les sociétés produisant des interfaces et extensions pour ce standard. Affaire à suivre.

Une grande évolution semble se dessiner en ce qui concerne les tableurs. Pendant dix ans, les enfants de VisiCalc, le premier du genre, se sont contentés de plagiat simple et peu inventif. Puis sont venus les intégrés, mais la partie calcul n'évoluait toujours pas. Il semble que maintenant, la créativité soit à nouveau présente dans ce secteur de l'activité logicielle. Parmi les nouveautés, il faut signaler le grand nombre de programmes qui permettent d'ajouter des possibilités aux programmes existants. Ainsi, Hal, de Lotus, qui complète 1-2-3 de la même société, ou Note It, de Turner Hall, de nouveaux venus sur le marché.

Hal est une interface «intelligente» pour 1-2-3, permettant d'ajouter des commandes, de gérer le tout avec une plus grande simplicité, ce qui n'est pas un mal, et facilitant le contact avec l'utilisateur. C'est sans doute la raison de son nom, celui que portait l'ordinateur intelligent et capable de raisonnement du film «2001, Odyssée de l'Espace». Rappelons que Hal, dans le film, se détraque et finit par tuer.

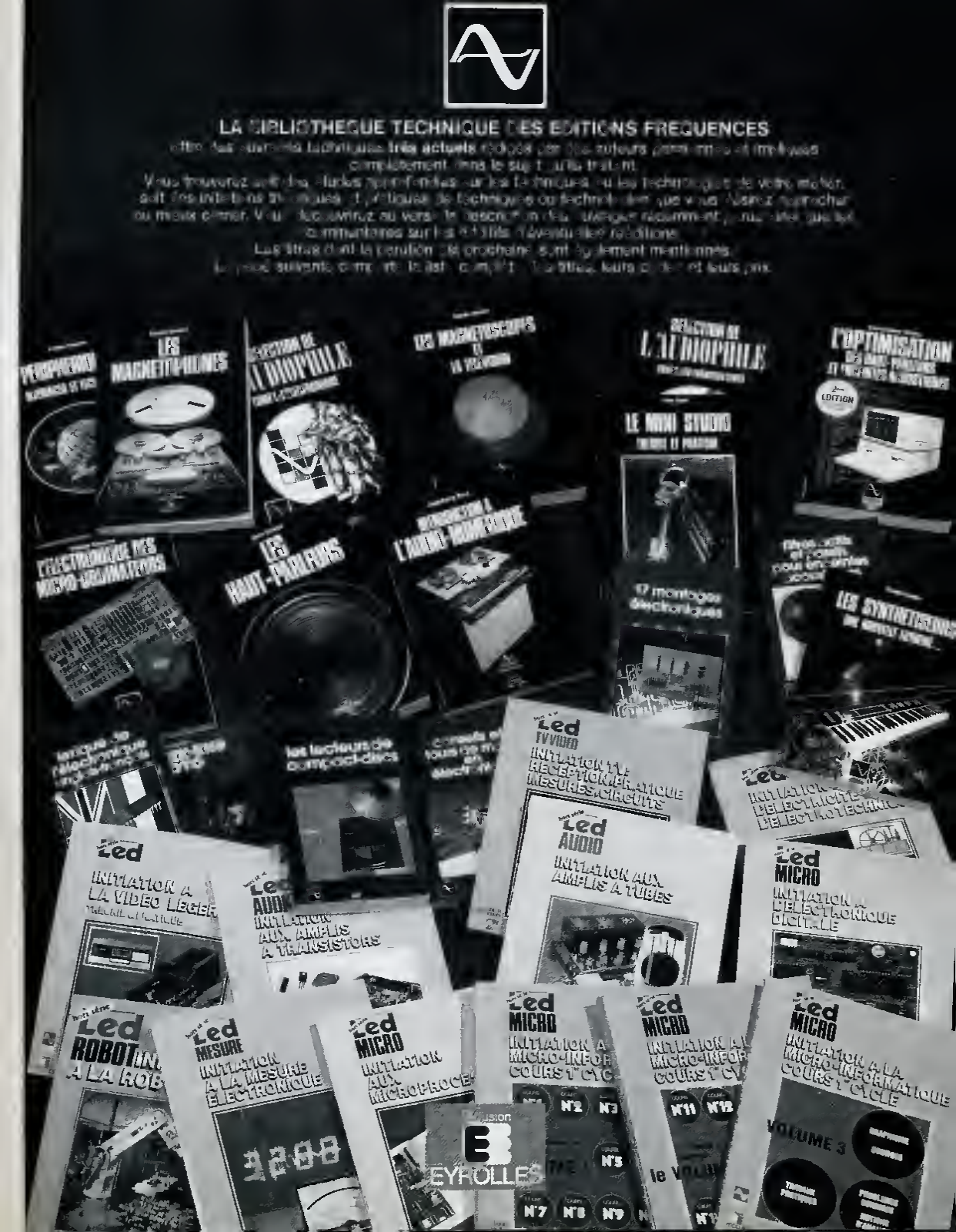
Mais il y a aussi des programmes qui proposent tout cela directement, plus bien d'autres choses encore, toutes bien nouvelles. Parmi ces programmes, le nouveau Silk, de Daybreak Technologies, ou Trapeze, ce dernier pour le Mac. Ces programmes permettent d'automatiser un grand nombre de calculs, en propageant des formules à l'aide d'une simple touche, de surveiller l'évolution de certains résultats tout au long de l'élaboration de la feuille de calculs, ou de simuler ces résultats à l'aide d'une feuille spéciale, destinée à des essais numériques. Il est possible de choisir un résultat, puis de laisser le programme trouver les meilleures données de départ permettant de réellement obtenir la solution «imposée». Donc, un tableur devient un concurrent direct de programmes de simulations, ou de gestions numériques. Silk permet même de modifier (un peu) la structure des données pour récupérer sous un format différent des fichiers venant d'un gros système. Il est aussi possible de définir des structures plus complexes que des nombres, par exemple des couples de nombres, des tableaux, et autres formes utiles dans divers types d'applications. Il est possible de pister les problèmes par suivi des calculs, ce qui permet d'éliminer la bête noire de ces programmes, les références circulaires, c'est-à-dire des cases que l'on calcule à partir d'autres, ces dernières ayant besoin des premières, ce qui arrive sur les grosses feuilles, ou il arrive que la logique des enchaînements soit délicate à décortiquer et où toute modification devient délicate à effectuer. Ces programmes permettent aussi de sauver en permanence sur disque les commandes tapées et si un problème fait planter le système, il est possible de reprendre où l'on en était. Enfin, Trapeze, tirant parti des possibilités de graphisme du Mac, permet de créer des cases ou des zones graphiques. C'est là une amélioration formidable, car avant, il fallait récupérer les résultats du tableur, les mettre dans un traitement de texte ou de graphiques, arranger le tout, puis imprimer. Avec Trapeze, tout peut être fait d'un seul coup. C'est en particulier plus puissant qu'un logiciel intégré, où tout cohabite mais, en France, vous savez maintenant que cohabiter n'est pas toujours une panacée.

Un utilisateur de programme comme vous et moi en a eu assez, un jour, de se battre avec. Las de chercher en vain après avoir essayé dix tableurs et une bonne vingtaine de traitements de textes, il vient de publier une sorte de code du programmeur. Sept points à retenir :

- pas de protection,
- pas de programme d'installation de programme, ces programmes qui permettent de configurer le logiciel à votre système. Au logiciel de s'adapter,
- Mise à jour des parties interfaces du logiciel par des commandes du programme et non par un programme annexe. Ceci permet de modifier les spécifications sans sortir du programme,
- Indépendance des gestions d'entrées-sorties vis-à-vis du programme, et ajout dans la documentation du programme de la description des commandes permettant de créer de nouvelles entrées-sorties pour des périphériques à venir,
- Un guide référence complet, c'est-à-dire un livre séparé réservé à la description de chaque commande,
- Aide à l'écran en permanence, avec adaptation automatique à la situation du moment,
- Interface systématique pour une souris pour les ordinateurs qui ne la proposent pas en standard.

Avec toutes ces propositions, un programme pourra devenir un vrai plaisir, et non plus un cauchemar de l'utilisateur occasionnel. Reste à savoir si les développeurs suivront le chemin qui semble pourtant s'imposer. Le passé proche autorise à l'optimisme.

Au mois prochain.


LA BIBLIOTHEQUE TECHNIQUE DES EDITIONS FREQUENCES
Led
INITIATION A LA VIDEO LEGERE
INITIATION AUX AMPLIS A TRANSISTORS
INITIATION AUX AMPLIS A TUBES
INITIATION A L'ELECTRONIQUE DIGITALE
INITIATION A LA MESURE ELECTRONIQUE
INITIATION AUX MICROPROCESSEURS
INITIATION A LA MICRO-INFORMATIQUE
COURS 1er CYCLE
VOLUME 3
EYROLLES

VIENT DE PARAITRE :

- **11 auteurs**
- **360 pages**
- **300 schémas et illustrations**

Prix : 350 F

Il y a bientôt trois ans démarrait ce travail de fond. Plus de vingt auteurs étaient sollicités pour concentrer en trois tomes les techniques du son. Le premier tome vient de paraître. Il traite de l'**Acoustique fondamentale**, des **Sources acoustiques**, de l'**Acoustique architecturale**, de la **Perception auditive**, des **Notions fondamentales de l'Electricité**, de l'**Enregistrement magnétique** ainsi que de la **Technologie audio-numérique.**

L'équipe des plus grands spécialistes actuels a été animée par Denis Mercier. Ensemble, ils ont mis sur pied un ouvrage actuellement unique au monde.

PROCHAINEMENT :

Collection jaune **Etude autour du 6809 (constructions et logiciels)** de Claude Vicidomini
L'image numérique de Jean-Marc Nasr
Le Basic structuré de Jean-François Coblentz
Divertissements en Basic de Franck Brown

Collection noire **La création musicale par ordinateur** de Frederic Leve
Pratique de l'Amiga de Henri Cohen et François Dress

Collection noire (format 165 × 240)

	Réf	Prix TTC
LES SYNTHETISEURS, UNE NOUVELLE LUTHERIE de Claude Gendre 154 p — Face au développement spectaculaire des synthétiseurs, grâce à l'électronique numéri que, le besoin d'un ouvrage complet accessible et surtout bien informé des dernières ou futures techniques se faisait ressentir. Le vœu est comblé en 180 pages	E 15	140 F
LES HAUT-PARLEURS de Jean Hiraga 320 p — Un gros volume qui connaît un succès constant : bien plus qu'un traité, il s'agit d'une véritable encyclopédie alliant théorie et pratique, histoire et une mine inépuisable d'informations reconnue dans le monde entier	E 01	185 F
INTRODUCTION A L'AUDIO-NUMERIQUE de Jean Pierre Picot 160 p — C'est le premier ouvrage paru en langue française traitant de l'audio numérique, écrit par un professionnel, avec rigueur et simplicité. L'explique brillamment les bases de cette technique : quantification, conversion, formats, codes d'erreurs	E 05	155 F
L'OPTIMISATION DES HAUT-PARLEURS ET ENCEINTES ACOUSTIQUES de Charles-Henry Delaleu — 240 p — Seconde édition améliorée d'un ouvrage fort attendu des passionnés d'électroacoustique. Ce livre permet aux amateurs et aux professionnels de se familiariser avec les rigoureuses techniques de modélisation des haut-parleurs et enceintes acoustiques et d'en mener à bien la réalisation	E 04	154 F
LES MAGNETOPHONES de Claude Gendre — 160 p — Pour tout savoir sur le magnétophone depuis l'avènement de cette mémoire des temps modernes, jusqu'aux enregis treurs numériques, en passant par la cassette. «Les magnétophones» est un ouvrage pratique, complet, indispensable à l'amateur d'enregistrement magnétique	E 02	92 F
LES MAGNETOSCOPES ET LA TELEVISION de Claude Gendre 256 p — Complément direct des «Magnétophones», «Les magnétoscopes et la télévision» débute par un bref histor que de la télévision et la description des premiers magnétoscopes. La théorie et la pratique de la capture et de l'enregistrement moderne des images vidéo en sont la teneur essentielle	E 03	155 F
L'ELECTRONIQUE DES MICRO-ORDINATEURS de Philippe Faugeras — 128 p — Cet ouvrage est destiné aux électroniciens désireux d'aborder l'étude d'un «hard» des micro-ordinateurs. Cette étude s'articule autour du microprocesseur Z 80 très répandu, et en décrit les éléments périphériques : mémoire, clavier, écran, interfaces de toutes sortes	E 08	150 F
PERIPHERIQUES : INTERFACES ET TECHNOLOGIE de Philippe Faugeras — 136 p — Faisant suite à la parution de «L'électronique des micro-ordinateurs», cet ouvrage s'adresse aux électroniciens désireux de s'initier aux montages périphériques des micro-ordinateurs : interfaces en particulier, qui permettent la communication avec le monde extérieur	E 22	150 F
SELECTION DE L'AUDIOPHILE - TOME 1 : L'ELECTRONIQUE 256 p	E 13	155 F
SELECTION DE L'AUDIOPHILE - TOME 2 : LES TRANSDUCTEURS 256 p Introuvable aujourd'hui, une sélection des meilleurs articles de la célèbre revue «L'Audiophile». Le tome 1 traite de l'électronique audio à tubes et transistors. Dans un esprit identi que, le tome 2 traite du domaine passionnant que constituent les transducteurs en audio	E 12	165 F
LE MINI STUDIO de Denis Fortier — 160 p — Le monde de l'audio évolue. Un secteur d'activité entièrement neuf vient d'apparaître : les mini studios. L'ouvrage de Denis Fortier, ingénieur du son, aborde le sujet de la manière la plus globale. Après les données physiques indispensables, le choix des maîtres, la manière d'installer et d'exploiter	E 25	140 F
LES TECHNIQUES DU SON Collectif d'auteurs sous la direction de Denis Mercier — 360 p — Le Livre des Techniques du Son est le premier ouvrage interdisciplinaire en langue française s'adressant aux professionnels du son	E 33	350 F

Collection rouge (format 135 × 210)

	Réf	Prix TTC
CONSEILS ET TOURS DE MAIN EN ELECTRONIQUE de Jean Hiraga 160 p — Le «dernier coup de patte» apporté à un montage, celui qui fait la différence entre le réali sation approximative et le kit bien fini, ce savoir faire s'acquiert au fil des ans — ou en parcourant «Conseils et tours de main en électronique»	L 07	88 F
LES LECTEURS DE COMPACT-DISCS 200 p — Tout beau, tout nouveau, le lecteur laser. Qu'en est-il réellement ? Pour en savoir plus, un livre traitant du sujet s'imposait. «Les lecteurs de compact discs» permet de faire son choix parmi 37 modèles testés, analysés, examinés et écoutés	L 10	130 F
LEXIQUE DE L'ELECTRONIQUE ANGLAIS-FRANÇAIS de Jean Hiraga - 72 p — Pour la première fois en électronique, un lexique anglais-français est présenté sous une forme pratique avec en plus des explications techniques, succinctes mais précises. Ce sont plus de 1 500 mots ou termes anglais qui n'auront plus de secret pour vous	L 09	85 F
FILTRES ACTIFS ET PASSIFS POUR ENCEINTES ACOUSTIQUES de Charles Henry Delaleu — 160 p — Finis les calculs fastidieux et erronés ! Grâce à cet ouvrage, les concepteurs d'enceintes acoustiques gagneront un temps appréciable durant la phase d'étude et de mise au point : 120 abaques et tableaux pour tous types de filtres si d'impédances de HP !	L 11	85 F
17 MONTAGES ELECTRONIQUES de Bernard Duval 128 p — Voici enfin réunies dans un même ouvrage, dix sept descriptions complètes et précises de montages électro niques simples. Il s'agit de réalisations à la portée de tous, dont don nombre d'exemplaires (sinon plus) fonctionnent régulièrement. Les schémas d'implantation et de circuits imprimés sont systématiquement publiés	L 14	95 F
WEEK-END PHOTO de Philippe Folie Dupart 208 p — Accessible à tous, «Week-end photo» permet de découvrir de façon simple les différents aspects de la photographie et de la belle. Vous y trouverez les bases indispensables pour vous perfectionner, un guide de choix des appareils 24 × 36 et des illustrations abondamment commentées	L 20	130 F

Collection jaune (format 210 × 270)

	Réf	Prix TTC
INITIATION A LA ROBOTIQUE 96 p — Cet ouvrage eut un succès retentissant dès sa sortie. Bien plus qu'un cours d'initiation, il s'agit aussi du premier recueil d'informations données par les concepteurs, les utilisateurs et les fans de cybernétique que nous recevons	(épuisé)	
INITIATION A LA MICRO-INFORMATIQUE COURS 1er CYCLE - LE VOLUME 1 de Claude Poigar 272 p	P 08	115 F
INITIATION A LA MICRO-INFORMATIQUE COURS 1er CYCLE - LE VOLUME 2 de Claude Poigar 208 p	P 16	130 F
INITIATION A LA MICRO-INFORMATIQUE COURS 1er CYCLE - LE VOLUME 3 de Claude Poigar 250 p Passe les premiers émois de la découverte de l'ordinateur, le moment est venu de tenter d'en réunir les renseignements. Une lacune apparaît. Celle d'un ouvrage d'initiation à la programmation, un vrai et complet	P 17 P 27	130 F 190
INITIATION A L'ELECTRONIQUE DIGITALE de Philippe Duquesne 104 p — Ce cours d'initiation à l'électronique digitale est dû à Ph. Duquesne, chargé de cours de microprocesseurs au CNAM. L'objet de ce cours âgé est de présenter les opérateurs logiques et leurs associations. La technologie est évoquée brièvement mais aussi	P 19	95 F
INITIATION AUX MICROPROCESSEURS de Philippe Duquesne 136 p — Du même auteur Ph. Duquesne, on nous propose cette fois-ci de pénétrer au cœur même de l'ordinateur, de comprendre le fonctionnement de l'élément vital qu'est le microprocesseur et enfin de maîtriser l'assembleur, langage du microprocesseur	P 18	95 F
INITIATION TV : RECEPTION, PRATIQUE MESURES, CIRCUITS de Roger-Charles Houzé 136 p — Issu d'un cours régulièrement remis à jour, ce livre permet à l'amateur comme au professionnel de se tenir au courant de l'état actuel de la technologie en télévision. De nombreux schémas explicatifs illustrent le contenu du livre	P 21	135 F
INITIATION A LA MESURE ELECTRONIQUE de Michel Casabo 120 p — Il n'existait pas jusqu'à présent un ouvrage couvrant de manière générale mais précise l'ensemble des problèmes relatifs à l'instrumentation et à la méthodologie du laboratoire électronique. C'est chose faite aujourd'hui avec ce volume récemment paru	P 23	140 F
INITIATION AUX AMPLIS A TRANSISTORS de Gilles Le Doré 96 p — Après un bref historique du transistor, cet ouvrage traite essentiellement de la conception des amplificateurs modernes à transistors. La théorie est décrite de manière simple et abordable, illustrée d'exemples de réalisations commerciales. Le but du livre est de donner à chacun la possibilité de réaliser soi-même son amplificateur	P 24	130 F
INITIATION AUX AMPLIS A TUBES de Jean Hiraga 152 p — Complémentaires des «Amplis à transistors», «les Amplis à tubes» sera certainement une petite encyclopédie sur ce sujet. Historique, mais aussi polémique puisque les tubes sont encore à actualité et parce que les arguments en faveur de cette technique et ses défenseurs sont encore nombreux	P 25	155 F
INITIATION A L'ELECTRICITE ET A L'ELECTROTECHNIQUE de Roger Friedenich 110 p — Vous trouverez aisément en librairie des ouvrages d'initiation à l'électronique ou aux techniques les plus avancées des circuits intégrés etc. Mais si vous désirez une initiation aux bases de l'électricité et de l'électrotechnique sans vous en remettre à des ouvrages scolaires, alors vous ne trouverez pas	P 28	150 F
INITIATION A LA VIDEO LEGERE - THEORIE ET PRATIQUE de Claude Gendre 72 p — Choix d'un standard ? Camescopes VHS, VHS C ou 8 mm ? Connexion ? Compatibilité ? Accessoires ? Montage ? Enfin, comment l'aimer ? Le nouveau livre de Claude Gendre répond à toutes ces questions. Cet ouvrage essentiellement pratique n'a pas d'équivalent en librairie aujourd'hui	P 29	100 F
LES MONTAGES ELECTRONIQUES de Jean Pierre Lemoine 276 p — Véritable encyclopédie. Plus de 1 000 dessins, 25 montages originaux	P 30	250 F
LE TELEPHONE ET LES RADIOTELEPHONES de Roger Charles Houzé 96 p, 73 schémas	P 31	130 F
LES BASES DE L'ELECTRONIQUE de Raymond Breton 84 p, 162 schémas. Vous ne connaissez pas l'électron que ce livre vous permet d'accéder aux bases nécessaires, mais néanmoins d'atteindre un niveau vous permettant d'aborder des constructions de bases	P 32	120 F

Diffusion auprès des libraires assurée exclusivement par les Editions Eyrolles.

Bon de commande à retourner aux Editions Fréquences 1, boulevard Ney 75018 Paris.

Je désire recevoir le(s) ouvrages(s) ci-dessous référencé(s) que je coche d'une croix :

							(épuisé)			
E 01 ☐	E 02 ☐	E 03 ☐	E 04 ☐	E 05 ☐	E 08 ☐	L 07 ☐	P 08 ☐	L 09 ☐	L 10 ☐
L 11 ☐	E 12 ☐	E 13 ☐	L 14 ☐	E 15 ☐	P 16 ☐	P 17 ☐	P 18 ☐	P 19 ☐	L 20 ☐
P 21 ☐	E 22 ☐	P 23 ☐	P 24 ☐	E 25 ☐	P 26 ☐	P 27 ☐	P 28 ☐	P 29 ☐	P 30 ☐
P 31 ☐	P 32 ☐	E 33 ☐							

Frais de port : + 12 F par livre commandé, soit la somme totale ci-jointe de Frs ____ par CCP ☐ Chèque bancaire ☐ Mandat-lettre ☐

Nom ____ Prénom ____

Adresse ____

Ville ____ Code Postal ____

ET LA TECHNIQUE ?

Si l'informatique a été, à l'origine, créée pour des besoins scientifiques, il faut bien admettre que les applications de gestion ont très nettement pris le dessus. Or depuis peu, il semble que nous arrivions vers une certaine saturation du marché. En effet, de gros clients sont déjà bien servis ! Ainsi les grands constructeurs reviennent-ils à leurs premières amours.

Finalement, si les bureaux sont équipés, le moins que l'on puisse dire, c'est que les ateliers sont sous-équipés. C'est bien connu. Une société de fabrication de matériel informatique ne peut faire des efforts que pour les marchés à fort potentiel de développement économique. Aussi, après quelques légers problèmes, nombreux sont les fabricants qui ont décidé de porter tous leurs efforts vers l'informatique industrielle. L'informatique peut s'implanter, dans une très large mesure, dans le cycle de fabrication d'un objet quelconque. Qu'il s'agisse de C.A.O., de C.F.A.O. ou de contrôle de la qualité en passant par la gestion des stocks, il y a beaucoup à faire dans ce domaine. Comme toujours, c'est le militaire et l'aérospatiale qui entraînent ce long processus. Il convient donc, en partant des connaissances acquises dans ces activités, de démocratiser les systèmes informatiques techniques pour les rendre plus abordables envers l'industrie classique. Il est évident que devant les modifications apportées par les marchés internationaux, le contrôle de la qualité et les coûts de revient sont devenus de véritables outils de guerre économique. Ce n'est que grâce à une forte technicité que les pays dits «développés» pourront continuer à garder leur avance. L'informatique industrielle est prête pour jouer son rôle.

dBase III

Charles-Henry Delaleu

Au hit-parade des progiciels vendus en France, dBase III est le premier de gestion de base de donnees DBase III suit en effet une brillante carriere Rappelons que dBase II fut un des premiers logiciels a être diffuse a une grande echelle Ses limitations et quelques problèmes de jeunesse furent corriges dans la version dBase III Aujourd'hui, dBase III Plus est une version amelioree de dBase III Elle autorise l'utilisation de fichiers en réseaux De plus, certains algorithmes de tri ont ete re-etudies afin d'optimiser la vitesse d'execution sur gros fichiers

En prologue, Ashton-Tate présente dBase III comme un systeme de gestion de base de donnees de type relationnel C'est exact dans le principe mais il faut se souvenir que nous sommes dans un environnement micro-informatique Si les possibilités sont enormes, attention aux capacités de la machine ainsi qu aux temps d'execution

dBase III peut être utilisé de trois manieres distinctes : soit en mode clavier simple par interrogation, soit en mode ASSIST par systeme de menu déroulant, soit en mode programme en utilisant le langage de programmation de dBase III

Afin de faciliter l'utilisation de dBase III, deux utilitaires ont ete conçus pour créer des applications de gestion de fichiers simples (dBoutils) en automatique

DBASE

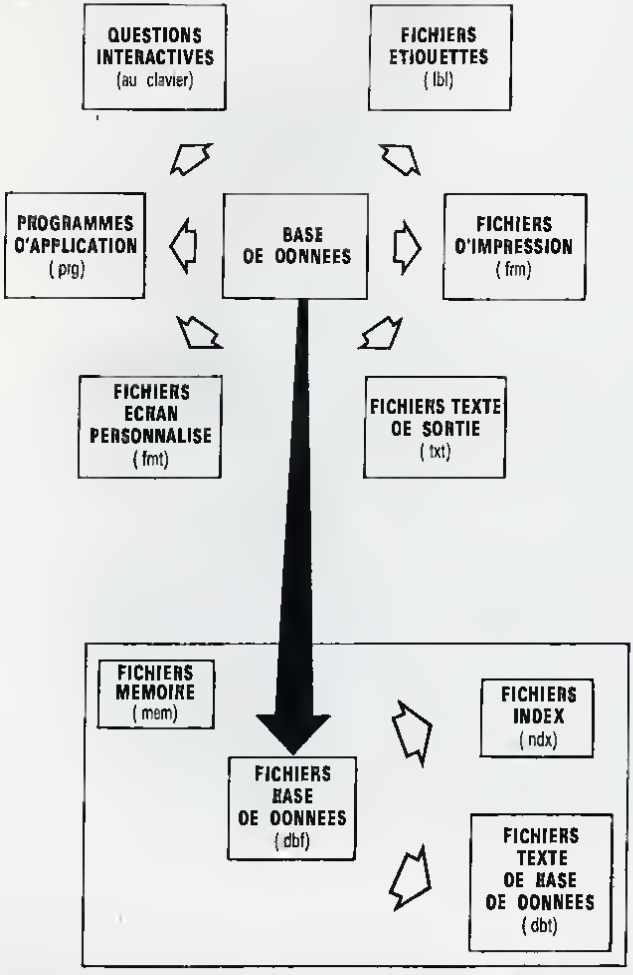
Dans cet article, nous ne presenterons pas dBase III Ceci a déjà ete fait dans notre n° 20 de mai 1985 De ce fait, nous traiterons surtout des differentes commandes La deuxieme partie de cet article sera consacree a un long exemple concernant la réalisation d'un fichier articles et sa gestion

RAPPEL DES PRINCIPALES CARACTERISTIQUES DE DBASE III

1. Système de gestion de base de données de type relationnel
2. Mode interactif ou mode programme
3. 128 champs et 4 000 caractères maximum par enregistrement
4. Nouveau type de données MEMO permettant aux enregistrements d'atteindre une longueur de plus de 500 000 caractères
5. Un milliard d'enregistrements par fichier
6. Utilisation possible et simultanée de 10 fichiers de données
7. Tri ultra-rapide sur plusieurs champs
8. Fichier indexé, recherche très rapide, etc.
9. Langage très complet, avec possibilité d'utiliser des procédures et des passages de paramètres

UN SYSTEME DE GESTION DE BASES DE DONNEES

dBase III est architecture autour d'un certain nombre de fichiers réalisant une base de données.



LES FICHIERS DBASE III

dBase III sauvegarde les informations sur des fichiers disques de neuf formats differents Chacun repond a un besoin particulier

a. Fichier base de données

Les fichiers de base de donnees stockent les informations Ils sont realises sous forme d enregistrements et de champs Chaque enregistrement contient un seul ensemble d'informations

b. Fichiers MEMO

Les fichiers MEMO sont des fichiers auxiliaires des fichiers de donnees Ils sont utilises pour stocker les informations de champs MEMO dBase III peut contenir un champ MEMO pour chaque enregistement Il s agit en fait d un texte se raccrochant a un enregistrement Ce texte peut servir a stocker des informations autres que les champs usuels

c. Fichiers index

Les fichiers index permettent d utiliser un fichier de donnees dans un ordre logique different de l ordre physique L ordre physique correspond a l ordre d enregistrement lors de la saisie L ordre logique peut être un ordre alphabetique ou numerique base sur le contenu d'un ou de plusieurs champs

d. Fichiers commande

Ils contiennent les programmes de gestion ou d application qui ont ete crees grâce a l editeur de dBase III Ils sont realises avec la commande MODIFY COMMAND

e. Fichiers format

Les fichiers format permettent de creer des ecrans de saisie personnalises et des elements de sortie personnalises sur imprimante

f. Fichiers étiquettes

Ceux-ci contiennent les informations necessaires pour remplir des etiquettes En effet, grâce a la commande LABEL il est possible d imprimer les informations sur un format d etiquette

g. Fichiers mémoire

Les fichiers memoire peuvent contenir jusqu a 256 variables memoire qui sont stockees dans une memoire tampon et peuvent être utilisees par plusieurs programmes ou applications a tout instant Il s agit, en d autres termes, de variables communes

h. Fichiers d'impression

Les fichiers d'impression contiennent les informations necessaires a la commande REPORT Cette derniere autorise la confection de formats d impression pre-determines, par exemple une edition de type facture ou formulaire pre-etabli

i. Fichiers texte

Les fichiers texte sont sous format ASCII Il s agit uniquement de caracteres imprimables Ils servent d'interface entre dBase III et d autres programmes utilisateurs

LA COMMANDE ASSIST

Parmi les trois modes d utilisation de dBase III, le mode Assist est tres interessant En effet, en mode simple (interrogation a partir du clavier), il convient de bien connaitre toutes les commandes de dBase III ainsi que leurs utilisations En mode programme, il faudra bien sûr ecrire ses programmes Le mode Assist simplifie la tâche Il pourra être utilise pour des applications simples Dans ce cas, l'utilisation des fichiers se fait en mode interactif a l aide de menus deroutants De plus en plus de progiciels utilisent ce mode de fonctionnement Le mode Assist est active en tapant ASSIST puis validation Chaque partie du logiciel possede un systeme de menus arborescents Le deplacement d un menu a l'autre et le pointage d une commande se font a l'aide des touches de deplacement du clavier (fleches) et de la touche validation (ENTER) Il est ainsi possible d'acceder à toutes les commandes et autres fonctions

LES PRINCIPALES COMMANDES DE DBASE III

USE	La commande USE indique a dBase III le fichier que l on desire utiliser Ex : USE NOMS : utiliser le fichier NOMS
DISPLAY	Cette commande permet de visualiser le contenu du premier enregistrement

DISPLAY Next 4	Il s'agit d'une extension de la commande DISPLAY. Ceci signifie visualiser du premier enregistrement jusqu'au quatrième.
CLEAR	Efface l'écran
LIST	Lister tous les enregistrements.
DISPLAY au	Idem à LIST, mais ici il faudra appuyer à la fin de chaque page écran sur ENTER pour voir la page suivante
DISPLAY STRUCTURE	Cette commande indique la structure d'un fichier : – le nom et le type de champ – la taille de chaque champ – le nombre d'enregistrements – la date de la dernière mise à jour.
DISPLAY RECORD 10	Affiche le contenu de l'enregistrement n° 10
GOTO 22	Aller à l'enregistrement n° 22
QUIT	Quitter dBase III
HELP	Appel du menu d'aide
APPEND	Commande pour ajouter un enregistrement
EDIT 12	Cette commande permet d'aller à l'enregistrement n° 12 pour effectuer une modification. Un menu d'aide et de commandes apparaît en haut de l'écran.
BROWSE	Cette commande permet d'accéder en mode modification sur tout le fichier
DELETE RECORD 5	Détruire l'enregistrement n° 5 (destruction logique)
RECALL ALL	Rappeler les enregistrements détruits par DELETE.
PACK	Détruire définitivement les enregistrements détruits par DELETE. Il s'agit ici d'une destruction physique.
CLEAR ALL	Cette commande ferme tous les fichiers ouverts

CREATION D'UN FICHIER

La création d'un fichier peut se faire soit en mode ASSIST, soit grâce à la commande CREATE. Cette dernière appelle une routine de création qui contient tous les ordres pour la création d'un fichier.

OPERATEURS RELATIONNELS ET LOGIQUES

Opérateurs relationnels

=	égal à
<	inférieur à
>	supérieur à
<=	inférieur ou égal à
>=	supérieur ou égal à
< >	différent de

Opérateurs logiques

AND	et logique
OR	ou logique
NOT	non logique

La commande FOR

La commande FOR permet de donner une ou plusieurs conditions aux commandes DISPLAY et LIST.

Ex : 1. DISPLAY NOM FOR NOM < 'N'
Afficher les noms dont le premier caractère est supeneur à N.
2. DISPLAY NOM, LOYER FOR LOYER < = 8000
Afficher les noms et les loyers pour chaque enregistrement dont le champ loyer est inférieur ou égal à 8 000 F.

LE TRI

Le tri rapide se fait en deux temps. Il convient de réaliser un double au fichier que l'on veut trier. Puis, dans un second temps, le tri est effectué en partant des données du fichier maître vers les fichiers triés.

La commande est :

SORT ON < liste de champs > TO < nouveau fichier >

INDEXATION

Dans le cas d un fichier trie, le fait d ajouter des enregistrements a toutes les chances de mettre un terme au tri prealablement effectue Il existe pour cela une autre solution : l indexation. Un fichier d'indexation est un fichier qui reprend l'ordre dans lequel les enregistrements devront etre relus En d autres termes, un champ d'un enregistrement se verra affecter d un numero d ordre de classement. Ainsi, l'ensemble du fichier n'a-t-il pas besoin d être entierement reorganise a chaque modification Il y a donc le fichier principal et un fichier simplifie d'indexation
Commande :

```
USE NOMS
INDEX ON NOM TO FAMILLE
```

Cela signifie : - utiliser le fichier NOMS
– indiquer le champ NOM sur FAMILLE

RECHERCHE RAPIDE D'UNE INFORMATION (FIND)

Grâce a l indexation, il est possible de retrouver rapidement une information.
Commande :

```
SET INDEX TO FAMILLE
FIND ROUX
```

Cela signifie : – se positionner sur l'index FAMILLE
– chercher le nom ROUX

FONCTIONS SUR LES CHAINES DE CARACTERES

+	Concatenation de chaines ou de champs
LEN	Evaluation de la longueur d'une chaîne ou d'un champ
UPPER	Conversion de minuscules en majuscules
TRIM	Elimination des espaces de la fin de la chaîne

LA COMMANDE ERASE

Cette commande permet d'effacer un fichier.

REALISATION D'UN PROGRAMME

Grâce à l editeur de texte de dBase III, il est possible d'ecrire des programmes d application partant des fichiers dBase III
Le langage de programmation fourni avec le progiciel est assez puissant et bien structure.
En deuxieme partie de cet article, nous présentons un programme de gestion d adresses sous dBase III. La structure du fichier adresses est la suivante :

```
. USE ADRESSE
. DISPLAY STRUCTURE
Structure for database: C:ADRESSE.dbf
Number of data records:        0
Date of last update    : 01/01/80
Field  Field Name  Type        Width     Dec
    1  NOM         Character      20
    2  PRENOM      Character      30
    3  ADRESSE     Character      40
    4  CODE        Character       5
    5  VILLE       Character      30
    6  TEL         Character      20
    7  DIVERS      Character      70
** Total **                     216
```

Tous les champs sont de type caractere.
Le programme est architecture en deux parties :

- le programme principal,
- les procédures de gestion des enregistrements

Les commandes principales du programme sont

1 Creation d'un enregistrement
2. Modification d'un enregistrement
3 Annulation d un enregistrement
4 Recherche d'un enregistrement
5 Suivant
6 Precedent
7 Edition
8. Index
9 Quitter

PROGRAMME PRINCIPAL

```
* Nom.....: ADRESSE.prg
* But.....: Gestion du fichier ADRESSE
* Date....: Le 31/ 3/1987
* Auteur..: GENCODE (Gnrateur de programmes dBASE III)
SET COLOR TO 7/0,0/7,0
CLEAR
SET DELETE      ON
SET EXACT       ON
SET TALK        OFF
SET HEADING     OFF
SET SCOREBOARD OFF
SET ESCAPE      ON
IF .NOT. FILE('ADRESSE.DBF')
   ? 'Öáááááááááááááááááááááááááááááááááááááááááááááááááá¢'
   ? '"Vous devez au pralable crer la structure du fichier de donnes"'
   ? '"                                                              ".'
   ? '"Employez la commande ÛCREATE£ de dBASE pour gnrer la structure"'
   ? 'ááááááááááááááááááááááááááááááááááááááááááááááááááái'
   RETURN
ELSE
   IF .NOT. FILE('ADRESSE.NDX')
      USE ADRESSE
      INDEX ON nom TO ADRESSE
   ENDIF
ENDIF
SET COLOR TO 7/0
SET PROCEDURE TO ADRESSE.PRC
CLOSE DATABASE
USE ADRESSE INDEX ADRESSE
PUBLIC xnom,xprenom,xadresse,xcp,xville,xtel,xdivers
PUBLIC PRESENT
TEXT

NOM:                              PRENOM:


ADRESSE:

CODE POSTAL:            VILLE:

TEL:

DIVERS:
```

```
ENDTEXT
@ 22,00 SAY 'èeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee'
@ 22,39 SAY 'eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee£'
@ 23,00 SAY '¤'
@ 23,78 SAY '¤'
@ 24,00 SAY 'àeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee'
@ 24,39 SAY 'eeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee¥'
STORE 'N' TO present
SET COLOR TO 0/7
@ 23,03 SAY 'Crer'
@ 23,09 SAY 'Modifier'
@ 23,18 SAY 'Annuler'
@ 23,26 SAY 'Rechercher'
@ 23,37 SAY 'Suivant'
@ 23,45 SAY 'Prcdent'
@ 23,55 SAY 'Edition'
@ 23,63 SAY 'Index'
@ 23,69 SAY 'Quitter'
SET COLOR TO 7/0
@ 23,77 SAY ' '
STORE .T. TO boucle
DO WHILE boucle
   SET COLOR TO 0/7
   DO WHILE .T.
      STORE CHR(INKEY()) TO reponse
      IF reponse # CHR(0)
         EXIT
      ELSE
         @ 00,71 SAY TIME()
      ENDIF
   ENDDO
   IF AT(reponse,'CMARSPEIQ') = 0
      ?? CHR(7)
      LOOP
   ELSE
      SET COLOR TO 7/0
      DO CASE
         CASE reponse='C'
            SET COLOR TO 7/0
            @ 23,03 SAY 'Crer'
            SET COLOR TO 0/7
            DO efface
            DO store
            DO cree
            SET COLOR TO 0/7
            @ 23,03 SAY 'Crer'

         CASE reponse='M' .AND. present='O'
            SET COLOR TO 7/0
            @ 23,09 SAY 'Modifier'
            SET COLOR TO 0/7
            DO storage
            DO saisie
            SET COLOR TO 0/7
            @ 23,09 SAY 'Modifier'
         CASE reponse='A' .AND. present='O'
```

```
      SET COLOR TO 7/0
      @ 23,18 SAY 'Annuler'
      SET COLOR TO 0/7
      DO annule
      SET COLOR TO 0/7
      @ 23,18 SAY 'Annuler'
   CASE reponse='R'
      SET COLOR TO 7/0
      @ 23,26 SAY 'Rechercher'
      SET COLOR TO 0/7
      DO efface
      DO cherche
      SET COLOR TO 0/7
      @ 23,26 SAY 'Rechercher'
   CASE reponse='S' .AND. .NOT. EOF()
      SET COLOR TO 7/0
      @ 23,37 SAY 'Suivant'
      SET COLOR TO 0/7
      SKIP
      DO affiche
      SET COLOR TO 0/7
      @ 23,37 SAY 'Suivant'
   CASE reponse='P' .AND. .NOT. BOF()
      SET COLOR TO 7/0
      @ 23,45 SAY 'Prcdent'
      SET COLOR TO 0/7
      SKIP-1
      DO affiche
      SET COLOR TO 0/7
      @ 23,45 SAY 'Prcdent'
   CASE reponse='E'
      SET COLOR TO 7/0
      @ 23,55 SAY 'Edition'
      SET COLOR TO 0/7
      DO efface
      DO edite
      SET COLOR TO 0/7
      @ 23,55 SAY 'Edition'
   CASE reponse='I'
      SET COLOR TO 7/0
      @ 23,63 SAY 'Index'
      SET COLOR TO 0/7
      DO efface
      SET COLOR TO 0/7
      @ 00,00 SAY SPACE(79)
      @ 00,01 SAY 'Ractualisation du fichier en cours...
      PACK
      SET COLOR TO 0/7
      @ 23,63 SAY 'Index'
      SET COLOR TO 7/0
      @ 00,00 SAY SPACE(79)
   CASE reponse='Q'
      SET COLOR TO 7/0
      @ 23,69 SAY 'Quitter'
      CLOSE PROCEDURE
```

```
            CLOSE DATABASES
            SET COLOR TO 7/0
            CLEAR
            RETURN
         OTHERWISE
            ?? CHR(7)
      ENDCASE
      SET COLOR TO 7/0
      @ 23,77 SAY ' '
   ENDIF
ENDDO
```

BIBLIOTHEQUE DE PROCEDURES

```
* Nom.....: ADRESSE.prc
* But.....: Procdures du fichier ADRESSE
* Date....: Le 31/ 3/1987
* Auteur..: GENCODE (Gnrateur de programmes dBASE III)
PROCEDURE cree
   STORE 'N' TO present
   SET COLOR TO 7/0,7/0
   STORE SPACE( 15) TO xnom
   @  4, 4 GET xnom
   READ
   IF xnom=' '
      @  4, 4 SAY SPACE( 15)
      RETURN
   ELSE
      STORE UPPER(xnom) TO xnom
      CLEAR GETS
      DO store
      APPEND BLANK
      DO saisie
      STORE 'O' TO present
   ENDIF
   RETURN
PROCEDURE cherche
   STORE 'N' TO present
   SET COLOR TO 7/0,7/0
   STORE SPACE( 15) TO xnom
   @  4, 4 GET xnom
   READ
   IF xnom=' '
      @  4, 4 SAY SPACE( 15)
      RETURN
   ELSE
      STORE UPPER(xnom) TO xnom
      CLEAR GETS
      FIND '&xnom'
      IF .NOT. EOF()
         DO AFFICHE
         STORE 'O' TO PRESENT
      ELSE
         SET COLOR TO 0/7
         @ 00,00 SAY SPACE(79)
         @ 00,01 SAY 'FICHE INCONNUE: Pressez une touche...'
```

```
            SET CONSOLE OFF
            WAIT
            SET CONSOLE ON
            SET COLOR TO 7/0
            @ 00,00 SAY SPACE(79)
         ENDIF
      ENDIF
      RETURN
   PROCEDURE annule
      SET COLOR TO 0/7
      @ 00,00 SAY '  Etes-vous sur de vouloir supprimer cet'
      @ 00,40 SAY 'te fiche (O/N):                          '
      @ 00,55 SAY ' '
      STORE ' ' TO reponse
      SET CONSOLE OFF
      WAIT TO reponse
      SET CONSOLE ON
      STORE UPPER(reponse) TO reponse
      @ 00,56 SAY reponse
      IF reponse='O'
         @ 00,59 SAY 'Confirmez (O/N): '
         STORE ' ' TO reponse
         SET CONSOLE OFF
         WAIT TO reponse
         SET CONSOLE ON
         STORE UPPER(reponse) TO reponse
         @ 00,76 SAY reponse
         IF reponse='O'
            DELETE
            STORE 'N' TO present
            DO EFFACE
         ENDIF
      ENDIF
      SET COLOR TO 7/0
      @ 00,00 SAY SPACE(79)
      RETURN
   PROCEDURE edite
      STORE 'N' TO present
      @ 23,03 SAY SPACE(73)
      @ 23,03 SAY '(1)=Edition par REPORT  (2)=Edition par LABEL  (3)=Fin'
      @ 23,57 SAY '     Votre choix:
      DO WHILE .T.
         @ 23,75 SAY 'U'
         STORE INKEY() TO reponse
         IF reponse < 49 .OR. reponse > 51
            SET COLOR TO 0/7
            @ 23,75 SAY '±'
            SET COLOR TO 7/0
            LOOP
         ELSE
            STORE CHR(reponse) TO reponse
            @ 23,75 SAY reponse
            DO CASE
               CASE reponse = '1'
                  @ 23,03 SAY SPACE(73)
                  @ 23,03 SAY 'Donnez le nom du fichier REPORT: ±        ±'
```

```
            SET COLOR TO 7/0,7/0
            STORE SPACE(8) TO fichier
            @ 23,37 GET fichier
            READ
            CLEAR GETS
            IF fichier # ' '
               STORE TRIM(fichier)+'.frm' TO fichier
               IF .NOT. FILE('&fichier')
                  @ 23,03 SAY "Ce fichier n'existe pas ou ne se "
                  @ 23,36 SAY 'trouve pas dans le rpertoire courant...'
                  SET CONSOLE OFF
                  WAIT
                  SET CONSOLE ON
               ELSE
                  SET CONSOLE OFF
                  REPORT FORM &fichier TO PRINT NOEJECT
                  SET CONSOLE ON
               ENDIF
            ENDIF
            EXIT
         CASE reponse = '2'
            @ 23,03 SAY SPACE(73)
            @ 23,03 SAY 'Donnez le nom du fichier  LABEL: ±        ±'
            SET COLOR TO 7/0,7/0
            STORE SPACE(8) TO fichier
            @ 23,37 GET fichier
            READ
            CLEAR GETS
            IF fichier # ' '
               STORE TRIM(fichier)+'.lbl' TO fichier
               IF .NOT. FILE('&fichier')
                  @ 23,03 SAY "Ce fichier n'existe pas ou ne se "
                  @ 23,36 SAY 'trouve pas dans le rpertoire courant...'
                  SET CONSOLE OFF
                  WAIT
                  SET CONSOLE ON
               ELSE
                  SET CONSOLE OFF
                  LABEL FORM &fichier TO PRINT
                  SET CONSOLE ON
               ENDIF
            ENDIF
            EXIT
         CASE reponse = '3'
            EXIT
      ENDCASE
   ENDIF
ENDDO
SET COLOR TO 0/7
@ 23,03 SAY 'Crer0'
@ 23,09 SAY 'Modifier0'
@ 23,18 SAY 'Annuler0'
@ 23,26 SAY 'Rechercher0'
@ 23,37 SAY 'Suivant0'
@ 23,45 SAY 'Prcdent0'
```

```
      @ 23,55 SAY 'EditionÙ'
      @ 23,63 SAY 'IndexÙ'
      @ 23,69 SAY 'QuitterÙ'
      SET COLOR TO 7/0
      @ 23,77 SAY ' '
      RETURN
   PROCEDURE saisie
      SET COLOR TO 7/0,7/0
      @  4,38 GET xprenom
      @  7, 8 GET xadresse
      @  9,12 GET xcp
      @  9,29 GET xville
      @ 11, 4 GET xtel
      @ 13, 7 GET xdivers
      READ
      CLEAR GETS
      REPLACE nom         WITH xnom
      REPLACE prenom      WITH xprenom
      REPLACE adresse     WITH xadresse
      REPLACE cp          WITH xcp
      REPLACE ville       WITH xville
      REPLACE tel         WITH xtel
      REPLACE divers      WITH xdivers
      RETURN
   PROCEDURE affiche
      SET COLOR TO 7/0
      @ 13, 7 SAY divers
      @ 11, 4 SAY tel
      @  9,29 SAY ville
      @  9,12 SAY cp
      @  7, 8 SAY adresse
      @  4,38 SAY prenom
      @  4, 4 SAY nom
      IF nom<>' '
         STORE 'O' TO PRESENT
      ENDIF
      RETURN
   PROCEDURE efface
      SET COLOR TO 7/0
      @ 13, 7 SAY SPACE( 20)
      @ 11, 4 SAY SPACE( 11)
      @  9,29 SAY SPACE( 10)
      @  9,12 SAY SPACE(  5)
      @  7, 8 SAY SPACE( 20)
      @  4,38 SAY SPACE( 15)
      @  4, 4 SAY SPACE( 15)
      STORE 'N' TO present
      RETURN
   PROCEDURE store
      STORE SPACE( 15) TO xprenom
      STORE SPACE( 20) TO xadresse
      STORE 0 TO xcp
      STORE SPACE( 10) TO xville
      STORE SPACE( 11) TO xtel
      STORE SPACE( 20) TO xdivers
      RETURN
```

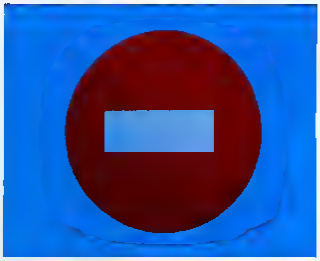
```
PROCEDURE storage
   STORE nom          TO xnom
   STORE prenom       TO xprenom
   STORE adresse      TO xadresse
   STORE cp           TO xcp
   STORE ville        TO xville
   STORE tel          TO xtel
   STORE divers       TO xdivers
   RETURN
```

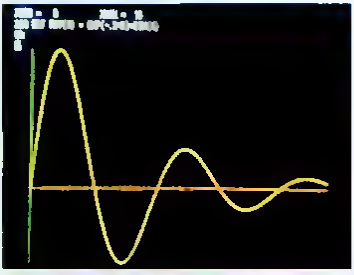
QUELQUES EXPLICATIONS SUR LES ORDRES UTILISES

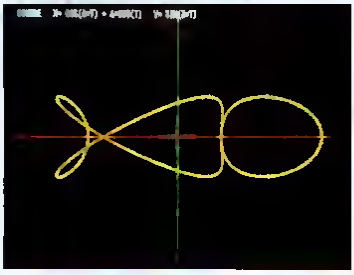
SET COLOR	Specifie les attributs d'ecran (couleur, noir et blanc, etc.)
SET DELETE	Affiche les enregistrements reperes pour effacement
SET EXACT	Exige une correspondance exacte lors d'une comparaison
SET TALK	Envoie le resultat de l'execution des commandes a l'ecran
SET HEADING	Affiche le nom des champs lors des commandes LIST et DISPLAY
SET ESCAPE	Continue l'execution d'un fichier de commande lorsque la touche ESC est enfoncee
CLOSE DATABASE	Ferme les fichiers
PUBLIC	Declare toutes les variables memoires globales
TEXT et ENDTEXT	Declare et termine une partie de texte
SAY	Localisation de l'affichage a l'ecran (ligne, colonne)
STORE	Archivage d'une donnee en memoire vive
DO WHILE	Faire tant que
CASE	Procedure de choix
DO	Executer un programme, une procedure Ex : DO EFFACE : executer la procedure Effacе
SKIP	Positionnement du pointeur en avant ou en arriere

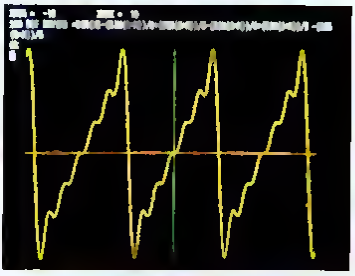
Initiation à la Micro-Informatique 1er Cycle Tome 3

(enfin paru !)









3.16 (Suite et fin) L'affichage

★ Etude des instructions permettant d'effectuer des présentations «évoluées» : PRINT TAB - PRINT USING - LOCATE - COLOR en mode texte
★ Presentation en tableaux de toutes sortes grace à la pratique des operateurs MODULO et DIVISION ENTIERE
★ Beaucoup de programmes utilisent des assemblages de ces instructions et operateurs : dont la combinaison n'est pas toujours facile

3.17 Compléments

★ Etude des dernieres instructions, fonctions et variables du cycle 1 : FILES, KILL, AUTO, ON ERROR GOTO, RESUME, ERR, ERL, DELETE, EDIT, RENUM TRON, TROFF, STOP, CONT, KEY ON, KEY OFF, FIX, BEEP
★ Compléments de cycle 1 qui sont maintenant accessibles aux élèves : sur la précision et les erreurs dues à l'arrondi, sur la sélection, les boucles

3.18 Graphisme

★ Une étude complete et détaillée sur les instructions graphiques en haute résolution : SCREEN, PSET, PRESET, STEP, LINE, CIRCLE, COLOR, POINT, PAINT, sans éluder aucune des difficultés et «pièges» classiques : l'incrustation de texte dans le dessin, les «bavures» dues au PAINT mal utilisé
★ Une étude detaillee du langage graphique DRAW, avec ses subtilites et ses pièges (sous-chaines X, parametres variables dans le DRAW, etc.)
★ De nombreux exercices avec leurs solutions (60) et leurs illustrations sur des photos d'ecran en couleur (48 photos)

3.19. Dessin des courbes

★ Un chapitre séparé du graphisme général (chapitre 3.18) de façon à ce que les «non matheux» puissent le sauter sans remords : ils ne seront pas punis !
★ Pour les matheux : une excellente revision et illustration des courbes de toutes sortes : Y = f (x), courbes parametrees, courbes en coordonnees polaires, avec des exemples utiles : courbes d'amortissement, astroide, cardioide, decomposition d'une fonction périodique par une série de Fourier

3.20. Révision générale

★ L'enchaînement des notions selon l'ordre «pedagogique» qui a ete utilise jusqu'ici est bien different de l'ordre «logique». Autant qu'un cours d'anglais suit un ordre different de celui (plus logique !) d'une grammaire anglaise
★ Tout ce qui a ete enseigne jusqu'ici resumé en 30 pages. Une reference pour retrouver la notion dont on a besoin a travers le cours et ses exercices. Mais aussi une reflexion sur la structure d'un langage informatique, d'ou une prépa ration a la lecture des cours de PASCAL (par exemple !)

3.21. Techniques de mise au point

★ Les outils de base : étude des éditeurs de texte, connaissance et interpretation des messages d'erreur
★ Comment rechercher et corriger ses erreurs
★ La representation du dialogue homme-machine, pour noter l'experience que vous acquerez par la pratique

3.22. Problèmes de synthèse - Notions d'analyse

C'est a la fois la conclusion, la partie la plus originale et la plus utile de ce cours. L'auteur ne se contente pas de fournir une liste de problèmes avec leur solution : il se met a la place du programmeur debutant en essayant de decortiquer le «processus de réflexion» qui fait passer de l'enonce d'un probleme a sa solution, une initiation pratique a l'analyse.

1 livre broché de 248 pages pages 21 × 27, dont 8 pages en couleur

nouveau !

- *exploiter toutes les possibilités des systèmes MIDI*
- *réaliser vous-mêmes un clip vidéo*
- *tirer le maximum de vos synthétiseurs*
- *installer chez vous votre studio d'enregistrement*
- *tout savoir sur les nouveautés musique et vidéo créatives*

Tout cela chaque mois dans Music Vidéo Systèmes

une publication des Editions Fréquences chez votre marchand de journaux

Editions Fréquences 1, boulevard Ney 75018 Paris - Tél. 45.07.01.07